

# A Unified Parallel Algorithm for Regularized Group PLS Scalable to Big Data

Pierre Lafaye de Micheaux, Benoît Liqueur and Matthew Sutton

**Abstract**—Partial Least Squares (PLS) methods have been heavily exploited to analyse the association between two blocs of data. These powerful approaches can be applied to data sets where the number of variables is greater than the number of observations and in presence of high collinearity between variables. Different sparse versions of PLS have been developed to integrate multiple data sets while simultaneously selecting the contributing variables. Sparse modelling is a key factor in obtaining better estimators and identifying associations between multiple data sets. The cornerstone of the sparsity version of PLS methods is the link between the SVD of a matrix (constructed from deflated versions of the original matrices of data) and least squares minimisation in linear regression. We present here an accurate description of the most popular PLS methods, alongside their mathematical proofs. A unified algorithm is proposed to perform all four types of PLS including their regularised versions. Various approaches to decrease the computation time are offered, and we show how the whole procedure can be scalable to big data sets.

**Index Terms**—Big data, High dimensional data, Partial Least Squares, Lasso Penalties, Sparsity, SVD.

## I. INTRODUCTION

IN this article, we review the *Partial Least Squares* (PLS) approach to big data. The PLS approach refers to a set of iterative algorithms originally developed by H. Wold [1], for the analysis of multiple blocks of data. This article focuses on PLS modelling when there are only two blocks of data. In the two blocks case, the PLS acronym (for Partial Least Squares or Projection to Latent Structures) usually refers to one of four related methods: (i) Partial Least Squares Correlation (PLSC) also called PLS-SVD [2]–[4], (ii) PLS in mode A (PLS-W2A, for Wold’s Two-Block, Mode A PLS) [5]–[7], (iii) PLS in mode B (PLS-W2B) also called Canonical Correlation Analysis (CCA) [8]–[10], and (iv) Partial Least Squares Regression (PLS-R, or PLS2) [11]–[13]. The first three methods model a symmetric relationship between the data, aiming to explain the shared correlation or covariance between the datasets, while the fourth method (PLS-R), models an asymmetric relationship, where one block of predictors is used to explain the other block.

P. Lafaye de Micheaux is with the School of Mathematics and Statistics, University of New South Wales, Sydney, Australia e-mail: lafaye@unsw.edu.au.

B. Liqueur is with the Laboratory of Mathematics and its Applications, University of Pau et Pays de L’Adour, UMR CNRS 5142, France and ARC Centre of Excellence for Mathematical and Statistical Frontiers and School of Mathematical Sciences at Queensland University of Technology, Brisbane, Australia, e-mail: b.liqueur@univ-pau.fr.

M. Sutton is with the ARC Centre of Excellence for Mathematical and Statistical Frontiers and School of Mathematical Sciences at Queensland University of Technology, Brisbane, Australia, e-mail: m5.sutton@hdr.qut.edu.au.

These methods are now widely used in many fields of science, such as genetics [14]–[16], neuroimaging [17], [18] and imaging-genetics [19], [20].

Recently, some authors have started to modify these methods using sparse modelling techniques; see e.g., [21]–[25]. These techniques refer to methods in which a relatively small number of covariates have an influence in the model. They are powerful methods in statistics that provide improved interpretability and better estimators, especially for the analysis of big data. For example, in imaging-genetics, sparse models demonstrated great advantages for the identification of biomarkers, leading to more accurate classification of diseases than many existing approaches [26].

In Section II, we survey the standard PLS methods. The optimization criteria and algorithmic computation is described. We pay particular attention to the singular value decomposition (SVD) due to its important role in the regularised PLS methods. Gathering in a single document an accurate description of all these methods, alongside with their complete mathematical proofs, constitutes a valuable addition to the literature; see also [7]. A link between the SVD of a matrix (constructed from deflated versions of the original matrices of data) and least squares minimisation in linear regression makes clear how to add regularization to obtain sparsity of the PLS results. This enables us to present in Section III the sparse versions of the four types of PLS, as well as a recent group version and a recent sparse group version. An unified algorithm is then presented in Section IV to perform all four types of PLS including regularised versions. Various approaches to decrease the computation time are proposed. We explain how the whole procedure can be made scalable to big data sets (any number of measurements, or variables). In Section V, we demonstrate the performance of the method on simulated data sets including the case of a categorical response variable. Our algorithm is implemented in the R programming language [27], and will be made available on the CRAN as a comprehensive package called `bigsgPLS` that includes parallel computations.

## II. PARTIAL LEAST SQUARES FAMILY

### A. Notation

Let  $\mathbf{X} : n \times p$  and  $\mathbf{Y} : n \times q$  be two data matrices (or “blocks”) both consisting of  $n$  observations of  $p$  and  $q$  variables respectively. These variables are generically noted  $X$  and  $Y$ . We assume from now on that these blocks are column-centered (since this turns matrix product into an estimate of covariance, up to a constant factor). Note that scaling is often recommended [13]. To make explicit the columns of a  $n \times r$

matrix  $\mathbf{A}$ , we write  $\mathbf{A} := [\mathbf{a}_1, \dots, \mathbf{a}_r] := (\mathbf{a}_j)$ . We also note  $\mathbf{A}_{\bullet h} := [\mathbf{a}_1, \dots, \mathbf{a}_h]$  for the submatrix of the first  $h$  columns ( $1 \leq h \leq r$ ), and  $\mathbf{A}_{\bullet \bar{h}} := [\mathbf{a}_{h+1}, \dots, \mathbf{a}_r]$  for the remaining ones. For two zero-mean vectors  $\tilde{\mathbf{u}}$  and  $\tilde{\mathbf{v}}$  of the same size, we note  $\text{Cov}(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}) = \tilde{\mathbf{u}}^\top \tilde{\mathbf{v}}$  and  $\text{Cor}(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}) = \tilde{\mathbf{u}}^\top \tilde{\mathbf{v}} / \sqrt{(\tilde{\mathbf{u}}^\top \tilde{\mathbf{u}})(\tilde{\mathbf{v}}^\top \tilde{\mathbf{v}})}$ . (The scaling factor  $(n-1)^{-1}$  is omitted w.l.o.g. for a reason that will be made obvious later on, and the tilde symbol is used to emphasize the fact that the vectors are not necessarily normed.) Let  $\mathbf{X}^+$  be the Moore-Penrose (generalized) inverse of  $\mathbf{X}$ . We note  $\mathcal{P}_{\mathbf{X}} = \mathbf{X}\mathbf{X}^+$  the orthogonal projection matrix onto  $\mathcal{I}(\mathbf{X})$ , the space spanned by the columns of  $\mathbf{X}$ , and  $\mathcal{P}_{\mathbf{X}^\perp} = \mathbf{I} - \mathcal{P}_{\mathbf{X}}$  the orthogonal projection matrix on the space orthogonal to  $\mathcal{I}(\mathbf{X})$ . When the inverse of  $\mathbf{X}^\top \mathbf{X}$  exists, we have  $\mathbf{X}^+ = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$ . The  $L_p$  vector norm ( $p = 1, 2$ ) of an  $n$ -length vector  $\mathbf{x}$ , is  $\|\mathbf{x}\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$ . The Frobenius norm of a  $n \times r$  matrix  $\mathbf{A}$  is  $\|\mathbf{A}\|_F = \|\text{vec}(\mathbf{A})\|_2$ , where the  $\text{vec}$  operator transforms  $\mathbf{A}$  into an  $nr \times 1$  vector by stacking its columns. The soft thresholding function is  $g^{\text{soft}}(x, \lambda) = \text{sign}(x)(|x| - \lambda)_+$ , where  $(a)_+ = \max(a, 0)$ . Finally,  $\otimes$  denotes the Kronecker product [28, (3), p. 662].

### B. Singular Value Decomposition

In all four PLS cases, the main linear algebra tool used is the singular value decomposition (SVD). For a real-valued matrix  $\mathbf{M} : p \times q$  of rank  $r$ , the (full) SVD is given by:

$$\mathbf{M} = \mathbf{U} \mathbf{\Delta} \mathbf{V}^\top = \sum_{l=1}^r \delta_l \mathbf{u}_l \mathbf{v}_l^\top, \quad (1)$$

where  $\mathbf{U} = (\mathbf{u}_l) : p \times p$  and  $\mathbf{V} = (\mathbf{v}_l) : q \times q$  are two orthogonal matrices whose columns contain the orthonormal left (resp. right) singular vectors, and  $\mathbf{\Delta} = \text{diag}(\delta_1, \dots, \delta_r, 0, \dots, 0) : p \times q$  is a rectangular matrix containing the corresponding ordered singular values  $\delta_1 \geq \delta_2 \geq \dots \geq \delta_r > 0$ .

Some properties of the SVD will be useful. First, for either orthogonal matrix  $\mathbf{U}$  or  $\mathbf{V}$  and any  $h = 1, \dots, r$ , we have

$$\mathbf{U}_{\bullet h}^\top \mathbf{U}_{\bullet h} = \mathbf{V}_{\bullet h}^\top \mathbf{V}_{\bullet h} = \mathbf{I}_h$$

where  $\mathbf{I}_h$  is the identity matrix. Note that in general  $\mathbf{U}_{\bullet h} \mathbf{U}_{\bullet h}^\top \neq \mathbf{I}_p$  unless  $h = p$ , and  $\mathbf{V}_{\bullet h} \mathbf{V}_{\bullet h}^\top \neq \mathbf{I}_q$  unless  $h = q$ . Second, for  $k < r$ , the SVD of  $\mathbf{M} - \sum_{l=1}^k \delta_l \mathbf{u}_l \mathbf{v}_l^\top$  is  $\sum_{l=k+1}^r \delta_l \mathbf{u}_l \mathbf{v}_l^\top$ . This is called the *SVD deflation property* and it will be used later on in an iterative manner.

Another important property of the SVD states that the (truncated) SVD of  $\mathbf{M}$  provides its best reconstitution (in a least squares sense) by a matrix with a lower rank ( $k$ , say) [29, Theorem 21.12.4]:

$$\left( \min_{\mathbf{A} \text{ of rank } k} \|\mathbf{M} - \mathbf{A}\|_F^2 \right) = \left\| \mathbf{M} - \sum_{l=1}^k \delta_l \mathbf{u}_l \mathbf{v}_l^\top \right\|_F^2 = \sum_{l=k+1}^r \delta_l^2.$$

If the minimum is searched over matrices  $\mathbf{A}$  of rank  $k = 1$ , where the matrix will be of the form  $\mathbf{A} = \tilde{\mathbf{u}} \tilde{\mathbf{v}}^\top$  (because all columns are multiples of one of the columns) and  $\tilde{\mathbf{u}}, \tilde{\mathbf{v}}$

are non-zero vectors (non necessarily normed, hence the tilde notation), we obtain

$$\min_{\tilde{\mathbf{u}}, \tilde{\mathbf{v}}} \left\| \mathbf{M} - \tilde{\mathbf{u}} \tilde{\mathbf{v}}^\top \right\|_F^2 = \left\| \mathbf{M} - \delta_1 \mathbf{u}_1 \mathbf{v}_1^\top \right\|_F^2 = \sum_{l=2}^r \delta_l^2.$$

Thus, solving

$$(\tilde{\mathbf{u}}_1, \tilde{\mathbf{v}}_1) = \underset{\tilde{\mathbf{u}}, \tilde{\mathbf{v}}}{\text{argmin}} \left\| \mathbf{M} - \tilde{\mathbf{u}} \tilde{\mathbf{v}}^\top \right\|_F^2 \quad (2)$$

gives us the first left and right singular vectors  $\mathbf{u}_1 = \tilde{\mathbf{u}}_1 / \|\tilde{\mathbf{u}}_1\|_2$  and  $\mathbf{v}_1 = \tilde{\mathbf{v}}_1 / \|\tilde{\mathbf{v}}_1\|_2$  of (1), as well as the first singular value  $\delta_1 = \|\tilde{\mathbf{u}}_1\|_2 \cdot \|\tilde{\mathbf{v}}_1\|_2$ . Note that this is also equivalent to solve

$$\underset{\|\mathbf{u}\|_2=1, \tilde{\mathbf{v}}}{\text{argmin}} \left\| \mathbf{M} - \mathbf{u} \tilde{\mathbf{v}}^\top \right\|_F^2 \quad \left( \text{resp. } \underset{\tilde{\mathbf{u}}, \|\mathbf{v}\|_2=1}{\text{argmin}} \left\| \mathbf{M} - \tilde{\mathbf{u}} \mathbf{v}^\top \right\|_F^2 \right)$$

followed by norming  $\tilde{\mathbf{v}}$  (resp.  $\tilde{\mathbf{u}}$ ).

### C. Penalised SVD

Shen and Huang [30] connected expression (2) to least squares minimisation in linear regression:

$$\begin{aligned} \left\| \mathbf{M} - \mathbf{u} \tilde{\mathbf{v}}^\top \right\|_F^2 &= \left\| \text{vec}(\mathbf{M}) - (\mathbf{I}_p \otimes \tilde{\mathbf{u}}) \tilde{\mathbf{v}} \right\|_2^2 \\ &= \left\| \text{vec}(\mathbf{M}) - (\mathbf{I}_q \otimes \tilde{\mathbf{v}}) \tilde{\mathbf{u}} \right\|_2^2. \end{aligned}$$

They present a method for sparse principal components by penalising the SVD as follows:

$$\underset{\|\mathbf{u}\|_2=1, \tilde{\mathbf{v}}}{\text{argmin}} \left\| \mathbf{M} - \mathbf{u} \tilde{\mathbf{v}}^\top \right\|_F^2 + P_\lambda(\tilde{\mathbf{v}}),$$

where  $\left\| \mathbf{M} - \mathbf{u} \tilde{\mathbf{v}}^\top \right\|_F^2 = \sum_{i=1}^p \sum_{j=1}^q (m_{ij} - u_i \tilde{v}_j)^2$  is the expanded Frobinus norm,  $P_\lambda(\tilde{\mathbf{v}})$  is a penalty function and  $\lambda \geq 0$  is a tuning parameter. After solving this problem, they calculate  $\mathbf{v} = \tilde{\mathbf{v}} / \|\tilde{\mathbf{v}}\|_2$ . Various forms for the penalisation term  $P_\lambda$  allow for different penalised variable selection techniques.

Following their idea, a number of PLS methods have been proposed based on an iterative algorithm. This algorithm has the basic form:

- ▷ Initialise  $\mathbf{u}$  and  $\mathbf{v}$  to have norm  $\|\mathbf{u}\| = \|\mathbf{v}\| = 1$ ;
- ▷ Solve

$$\underset{\tilde{\mathbf{v}}}{\text{argmin}} \left\| \mathbf{M} - \mathbf{u} \tilde{\mathbf{v}}^\top \right\|_F^2 + P_{\lambda_1}(\tilde{\mathbf{v}});$$

- ▷ Normalise  $\tilde{\mathbf{v}}$  to obtain  $\mathbf{v} = \tilde{\mathbf{v}} / \|\tilde{\mathbf{v}}\|$ ;
- ▷ Solve

$$\underset{\tilde{\mathbf{u}}}{\text{argmin}} \left\| \mathbf{M} - \tilde{\mathbf{u}} \mathbf{v}^\top \right\|_F^2 + P_{\lambda_2}(\tilde{\mathbf{u}});$$

- ▷ Normalise  $\tilde{\mathbf{u}}$  to obtain  $\mathbf{u} = \tilde{\mathbf{u}} / \|\tilde{\mathbf{u}}\|$ ;

where the penalty functions  $P_{\lambda_1}$  and  $P_{\lambda_2}$  enable us to obtain various sparse versions of the SVD. Applying this sparse SVD algorithm to the four standard PLS methods (i)–(iv) gives sparse PLS versions.

### D. Linking SVD to covariance and correlation

It is worthwhile recalling the close connection between SVD and maximum covariance (resp. maximum correlation) analyses; see Appendices A-A and A-B.

(C1): The values  $\mathbf{u}_h$  and  $\mathbf{v}_h$  ( $h = 1, \dots, r$ ) in (1) with  $\mathbf{M} = \mathbf{X}^\top \mathbf{Y}$  solve the minimisation problem

$$\begin{aligned} (\mathbf{u}_h, \mathbf{v}_h) &= \underset{\|\mathbf{u}\|_2 = \|\mathbf{v}\|_2 = 1, \delta > 0}{\operatorname{argmin}} \quad \|\mathbf{M} - \delta \mathbf{u} \mathbf{v}^\top\|_F^2 \\ &= \underset{\|\mathbf{u}\|_2 = \|\mathbf{v}\|_2 = 1}{\operatorname{argmax}} \quad \operatorname{Cov}(\mathbf{X} \mathbf{u}, \mathbf{Y} \mathbf{v}) \end{aligned}$$

subject to  $\mathbf{u}^\top \mathbf{u}_j = \mathbf{v}^\top \mathbf{v}_j = 0$ ,  $1 \leq j < h$ . We note that the vectors are unique up to changes in sign.

Note that the solution to this constrained optimization automatically satisfies:

$$\operatorname{Cov}(\mathbf{X} \mathbf{u}_h, \mathbf{Y} \mathbf{v}_j) = \mathbf{u}_h^\top \mathbf{X}^\top \mathbf{Y} \mathbf{v}_j = 0, \quad 1 \leq j < h$$

because the following matrix is diagonal

$$\mathbf{U}^\top \mathbf{M} \mathbf{V} = \mathbf{U}^\top \mathbf{U} \mathbf{\Delta} \mathbf{V}^\top \mathbf{V} = \mathbf{\Delta}.$$

(C2): Suppose that  $\mathbf{X}^\top \mathbf{X}$  and  $\mathbf{Y}^\top \mathbf{Y}$  are invertible. The solution to

$$(\tilde{\mathbf{w}}_h, \tilde{\mathbf{z}}_h) = \underset{\tilde{\mathbf{w}}, \tilde{\mathbf{z}}}{\operatorname{argmax}} \quad \operatorname{Cor}(\mathbf{X} \tilde{\mathbf{w}}, \mathbf{Y} \tilde{\mathbf{z}}), \quad h = 1, \dots, r,$$

subject to the constraints  $\operatorname{Cov}(\mathbf{X} \tilde{\mathbf{w}}, \mathbf{X} \tilde{\mathbf{w}}_j) = \operatorname{Cov}(\mathbf{Y} \tilde{\mathbf{z}}, \mathbf{Y} \tilde{\mathbf{z}}_j) = 0$ ,  $1 \leq j < h$  is given by  $\tilde{\mathbf{w}}_h = (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{u}_h$  and  $\tilde{\mathbf{z}}_h = (\mathbf{Y}^\top \mathbf{Y})^{-1/2} \mathbf{v}_h$ , where the  $\mathbf{u}_h$  and  $\mathbf{v}_h$  are found through (1) applied with  $\mathbf{M} = (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{X}^\top \mathbf{Y} (\mathbf{Y}^\top \mathbf{Y})^{-1/2}$ . Note that the  $\tilde{\mathbf{w}}_h$ 's (resp. the  $\tilde{\mathbf{z}}_h$ 's) are not necessarily orthonormal.

#### E. The four standard PLS methods

In this section, we survey the four standard PLS methods (i)–(iv) introduced in Section I. At its core, the four PLS methods are used to construct, iteratively, a small number  $H \leq r$  (chosen in practice using cross validation techniques) of meaningful linear combinations  $\xi_h = \mathbf{X} \mathbf{w}_h$  and  $\omega_h = \mathbf{Y} \mathbf{z}_h$  (or  $\xi_h = \mathbf{X} \tilde{\mathbf{w}}_h$  and  $\omega_h = \mathbf{Y} \tilde{\mathbf{z}}_h$  for un-normed weights) of the original  $X$ - and  $Y$ -variables, with either maximal covariance or correlation. These linear combinations are called component scores, or latent variables. Without additional constraints on the successive scores, there is only one solution for all methods, which is given by the first pair of singular vectors in either (C1) or (C2). So it is worthwhile noting that the various PLS methods impose additional *orthogonality* constraints on the optimisation, thus leading to the construction of multiple sets of component scores. Computationally, rather than finding component scores in terms of the original data with the required orthogonality, the PLS algorithms *deflate* the data matrices to ensure that solutions will have the required orthogonality. Component scores are then calculated using the modified (deflated) matrices, and are thus expressed at the  $h$ -th iteration as  $\xi_h = \mathbf{X}_{h-1} \mathbf{u}_h$  and  $\omega_h = \mathbf{Y}_{h-1} \mathbf{v}_h$  where  $\mathbf{X}_{h-1}$  and  $\mathbf{Y}_{h-1}$  are the deflated matrices.

The (normed) weights  $\mathbf{u}_h$  and  $\mathbf{v}_h$  are called the weight vectors (or direction vectors, or saliences, or effective loading weight vectors), while  $\mathbf{w}_h$  and  $\mathbf{z}_h$  (or  $\tilde{\mathbf{w}}_h$  and  $\tilde{\mathbf{z}}_h$  for un-normed versions) are called the adjusted weights. Since the adjusted weights define the score vectors in terms of the original data matrices (as opposed to the deflated matrices), the size of the elements of the weight vector can be interpreted

as the effect of the corresponding variables in the component score. On the other hand, the weight vectors  $\mathbf{u}_h$  and  $\mathbf{v}_h$  are defined in terms of the deflated matrices and cannot be interpreted this way.

The PLS algorithms can be seen as iterative methods that calculate quantities recursively using a deflation step to ensure appropriate orthogonality constraints. The construction of the components leads to decompositions of the original matrices  $\mathbf{X}$  and  $\mathbf{Y}$  of the form:

$$\mathbf{X} = \mathbf{\Xi}_H \mathbf{C}_H^\top + \mathbf{F}_H^X, \quad \mathbf{Y} = \mathbf{\Omega}_H \mathbf{D}_H^\top + \mathbf{F}_H^Y, \quad (3)$$

where  $\mathbf{\Xi}_H = (\xi_j)$  and  $\mathbf{\Omega}_H = (\omega_j)$  are called the  $X$ - and  $Y$ -scores,  $\mathbf{C}_H$  and  $\mathbf{D}_H$  are the  $X$ - and  $Y$ -loadings, and  $\mathbf{F}_H^X$  and  $\mathbf{F}_H^Y$  are the residual matrices.

We now detail the four classical cases (i)–(iv). We state the relevant PLS objective functions for the weight vectors  $\mathbf{u}_h$  and  $\mathbf{v}_h$  at each step  $h$ ,  $h = 1, \dots, H$ . We describe the deflation method in terms of deflating the matrices  $\mathbf{X}$  and  $\mathbf{Y}$  individually or deflating the matrix  $\mathbf{M} = \mathbf{X}^\top \mathbf{Y}$  directly, and the resulting orthogonality. We explicit all terms in the decomposition model (3). The relationship between the weight vectors  $\mathbf{u}_h$  and  $\mathbf{v}_h$  and the adjusted weights  $\mathbf{w}_h$  and  $\mathbf{z}_h$ , is given, as well as the PLS objective problem solved by the adjusted weights.

- (i) For PLS-SVD, the roles of  $\mathbf{X}$  and  $\mathbf{Y}$  are symmetric and the analysis focuses on modeling shared information (rather than prediction) as measured by the cross-product matrix  $\mathbf{R} = \mathbf{X}^\top \mathbf{Y}$ . Note that  $\mathbf{R}$  contains, up to some constant factor, the empirical covariances (resp. correlations) between  $X$ - and  $Y$ -variables when the columns of  $\mathbf{X}$  and  $\mathbf{Y}$  are centered (resp. standardised, in which case this method is sometimes called PLSC, for Partial Least Squares Correlation [2]).

The PLS-SVD objective function at step  $h$  is given by

$$(\mathbf{w}_h, \mathbf{z}_h) = \underset{\|\mathbf{w}\|_2 = \|\mathbf{z}\|_2 = 1}{\operatorname{argmax}} \quad \operatorname{Cov}(\mathbf{X} \mathbf{w}, \mathbf{Y} \mathbf{z}),$$

subject to the constraints  $\mathbf{w}^\top \mathbf{w}_j = \mathbf{z}^\top \mathbf{z}_j = 0$ ,  $1 \leq j < h$ . PLS-SVD searches for orthonormal directions  $\mathbf{w}_h$  and orthonormal directions  $\mathbf{z}_h$  ( $h = 1, \dots, H$ ), such that the score vectors  $\xi_h = \mathbf{X} \mathbf{w}_h$  and  $\omega_h = \mathbf{Y} \mathbf{z}_h$  have maximal covariance. Note that the scaling factor  $(n-1)^{-1}$  is omitted from the covariance (see Subsection II-A) and this has no impact on the argmax solution. Using (C1), the solutions to this problem are the  $H$  first columns of the matrices  $\mathbf{U}$  and  $\mathbf{V}$ , which are respectively the left and right singular vectors of  $\mathbf{M}_0 := (\mathbf{X}^\top \mathbf{Y})_0 := \mathbf{X}^\top \mathbf{Y}$ ; see (1). Another approach is to define  $\mathbf{u}_h = \mathbf{w}_h$ ,  $\mathbf{v}_h = \mathbf{z}_h$ ,  $\mathbf{X}_0 = \mathbf{X}$ ,  $\mathbf{Y}_0 = \mathbf{Y}$  and the deflated matrices  $\mathbf{X}_h = \mathbf{X}_{h-1} (\mathbf{I} - \mathbf{u}_h \mathbf{u}_h^\top) = \mathbf{X}_0 \prod_{j=1}^h (\mathbf{I} - \mathbf{u}_j \mathbf{u}_j^\top)$  and  $\mathbf{Y}_h = \mathbf{Y}_{h-1} (\mathbf{I} - \mathbf{v}_h \mathbf{v}_h^\top) = \mathbf{Y}_0 \prod_{j=1}^h (\mathbf{I} - \mathbf{v}_j \mathbf{v}_j^\top)$ . We have  $\mathbf{u}_h^\top \mathbf{X}_{h-1}^\top \mathbf{Y}_{h-1} \mathbf{v}_h = \mathbf{w}_h^\top \mathbf{X}^\top \mathbf{Y} \mathbf{z}_h$ . It is thus possible to replace the objective function with

$$(\mathbf{u}_h, \mathbf{v}_h) = \underset{\|\mathbf{u}\|_2 = \|\mathbf{v}\|_2 = 1}{\operatorname{argmax}} \quad \operatorname{Cov}(\mathbf{X}_{h-1} \mathbf{u}, \mathbf{Y}_{h-1} \mathbf{v})$$

and compute the previous scores as  $\xi_h = \mathbf{X}_{h-1} \mathbf{u}_h$  and  $\omega_h = \mathbf{Y}_{h-1} \mathbf{v}_h$ .

From (C1), and since  $\mathbf{X}$  and  $\mathbf{Y}$  are column-centered,

$$\text{Cov}(\boldsymbol{\xi}_h, \boldsymbol{\omega}_j) = 0, \quad j \neq h.$$

Note that the  $X$ - (resp.  $Y$ -) latent variables are not necessarily mutually orthogonal.

Now, because of the orthogonality properties on the  $\mathbf{u}_h$  and  $\mathbf{v}_h$ , we have

$$\begin{aligned} \mathbf{u}_h^\top \mathbf{X}^\top \mathbf{Y} \mathbf{v}_h &= \mathbf{u}_h^\top \left( \mathbf{X}^\top \mathbf{Y} - \sum_{l=1}^{h-1} \delta_l \mathbf{u}_l \mathbf{v}_l^\top \right) \mathbf{v}_h \\ &= \mathbf{u}_h^\top \mathbf{M}_{h-1} \mathbf{v}_h, \end{aligned}$$

where we define  $\mathbf{M}_h = \mathbf{X}^\top \mathbf{Y} - \sum_{l=1}^h \delta_l \mathbf{u}_l \mathbf{v}_l^\top = \mathbf{X}_h^\top \mathbf{Y}_h$ . It is thus possible to replace the previous optimization problem with

$$(\mathbf{u}_h, \mathbf{v}_h) = \underset{\|\mathbf{u}\|_2 = \|\mathbf{v}\|_2 = 1}{\operatorname{argmax}} \quad \mathbf{u}^\top \mathbf{M}_{h-1} \mathbf{v}.$$

The previous constraints are now automatically satisfied. Iterations (deflations) can be done using the relation  $\mathbf{M}_h = \mathbf{M}_{h-1} - \delta_h \mathbf{u}_h \mathbf{v}_h^\top$ . Thanks to the deflation property of the SVD, we have now that  $\delta_h$  is (resp.  $\mathbf{u}_h$  and  $\mathbf{v}_h$  are) the *first* singular value (resp. normed singular vectors) of  $\mathbf{M}_{h-1}$ .

Now, let  $\boldsymbol{\Xi}_H = \mathbf{X} \mathbf{U}_{\bullet H}$  and  $\boldsymbol{\Omega}_H = \mathbf{Y} \mathbf{V}_{\bullet H}$ . The decomposition model (3) is

$$\begin{aligned} \mathbf{X} &= \mathcal{P}_{\boldsymbol{\Xi}_H} \mathbf{X} + \mathcal{P}_{\boldsymbol{\Xi}_H^\perp} \mathbf{X} \\ &= \boldsymbol{\Xi}_H \boldsymbol{\Xi}_H^\top \mathbf{X} + \mathcal{P}_{\boldsymbol{\Xi}_H^\perp} \mathbf{X} \\ &= \boldsymbol{\Xi}_H \mathbf{C}_H^\top + \mathbf{F}_H^X \end{aligned}$$

and

$$\begin{aligned} \mathbf{Y} &= \mathcal{P}_{\boldsymbol{\Omega}_H} \mathbf{Y} + \mathcal{P}_{\boldsymbol{\Omega}_H^\perp} \mathbf{Y} \\ &= \boldsymbol{\Omega}_H \boldsymbol{\Omega}_H^\top \mathbf{Y} + \mathcal{P}_{\boldsymbol{\Omega}_H^\perp} \mathbf{Y} \\ &= \boldsymbol{\Omega}_H \mathbf{D}_H^\top + \mathbf{F}_H^Y, \end{aligned}$$

with  $\mathbf{C}_H = (\boldsymbol{\Xi}_H^\top \mathbf{X})^\top$  and  $\mathbf{D}_H = (\boldsymbol{\Omega}_H^\top \mathbf{Y})^\top$ .

(ii) For PLS-W2A, the optimisation problem at step  $h$  is

$$(\mathbf{u}_h, \mathbf{v}_h) = \underset{\|\mathbf{u}\|_2 = \|\mathbf{v}\|_2 = 1}{\operatorname{argmax}} \quad \text{Cov}(\mathbf{X}_{h-1} \mathbf{u}, \mathbf{Y}_{h-1} \mathbf{v})$$

where the deflated versions of the  $\mathbf{X}$  and  $\mathbf{Y}$  matrices are defined by  $\mathbf{X}_0 := \mathbf{X}$ ,  $\mathbf{Y}_0 := \mathbf{Y}$ ,

$$\begin{aligned} \mathbf{X}_h &:= \mathcal{P}_{\boldsymbol{\xi}_h^\perp} \mathbf{X}_{h-1} = \left( \prod_{j=h}^1 \mathcal{P}_{\boldsymbol{\xi}_j^\perp} \right) \mathbf{X} \\ &= \left[ \mathbf{I} - \boldsymbol{\xi}_h (\boldsymbol{\xi}_h^\top \boldsymbol{\xi}_h)^{-1} \boldsymbol{\xi}_h^\top \right] \mathbf{X}_{h-1} \end{aligned}$$

and

$$\begin{aligned} \mathbf{Y}_h &:= \mathcal{P}_{\boldsymbol{\omega}_h^\perp} \mathbf{Y}_{h-1} \\ &= \left[ \mathbf{I} - \boldsymbol{\omega}_h (\boldsymbol{\omega}_h^\top \boldsymbol{\omega}_h)^{-1} \boldsymbol{\omega}_h^\top \right] \mathbf{Y}_{h-1}, \end{aligned}$$

and where  $\boldsymbol{\xi}_h = \mathbf{X}_{h-1} \mathbf{u}_h$  and  $\boldsymbol{\omega}_h = \mathbf{Y}_{h-1} \mathbf{v}_h$ . These score vectors are stored in the matrices  $\boldsymbol{\Xi}_H = (\boldsymbol{\xi}_j)$  and  $\boldsymbol{\Omega}_H = (\boldsymbol{\omega}_j)$ .

We have  $\mathbf{X}_h = (\mathbf{I} - \mathcal{P}_{\boldsymbol{\xi}_h}) \mathbf{X}_{h-1} = (\mathbf{I} - \mathcal{P}_{\boldsymbol{\xi}_h})(\mathbf{I} - \mathcal{P}_{\boldsymbol{\xi}_{h-1}}) \mathbf{X}_{h-2} = (\mathbf{I} - \mathcal{P}_{\boldsymbol{\xi}_{h-1} : \boldsymbol{\xi}_h}) \mathbf{X}_{h-2}$  using [31, Theorem 7, p. 151]. Pursuing this argument leads to  $\mathbf{X}_h = \mathcal{P}_{\boldsymbol{\Xi}_{\bullet h}^\perp} \mathbf{X}$ , and similarly  $\mathbf{Y}_h = \mathcal{P}_{\boldsymbol{\Omega}_{\bullet h}^\perp} \mathbf{Y}$ . Now  $\boldsymbol{\xi}_h = \mathbf{X}_{h-1} \mathbf{u}_h = \mathcal{P}_{\boldsymbol{\Xi}_{\bullet h-1}^\perp} \mathbf{X} \mathbf{u}_h$  is clearly orthogonal to  $\boldsymbol{\xi}_j$  for  $j = 1, \dots, h-1$ . This argument clearly shows that

$$\text{Cov}(\boldsymbol{\xi}_h, \boldsymbol{\xi}_j) = \text{Cov}(\boldsymbol{\omega}_h, \boldsymbol{\omega}_j) = 0, \quad 1 \leq j < h.$$

PLS-W2A thus searches for successive  $X$ -score vectors (resp.  $Y$ -score vectors) that are orthogonal to the previous ones. The first pair  $(\boldsymbol{\xi}_1, \boldsymbol{\omega}_1)$  of  $X$ - and  $Y$ -score vectors is the one with maximal covariance. The next pairs are searched for using successively deflated (i.e., after removing the information contained in the previous pairs of scores) versions of  $\mathbf{X}_0$  and  $\mathbf{Y}_0$ . We can always write

$$\mathbf{X} = \mathcal{P}_{\boldsymbol{\Xi}_H} \mathbf{X} + \mathcal{P}_{\boldsymbol{\Xi}_H^\perp} \mathbf{X} \text{ and } \mathbf{Y} = \mathcal{P}_{\boldsymbol{\Omega}_H} \mathbf{Y} + \mathcal{P}_{\boldsymbol{\Omega}_H^\perp} \mathbf{Y}.$$

Thus, the elements of the decomposition model (3) are

$$\mathbf{C}_H = \mathbf{X}^\top \boldsymbol{\Xi}_H (\boldsymbol{\Xi}_H^\top \boldsymbol{\Xi}_H)^{-1}, \quad \mathbf{F}_H^X = \mathbf{X}_H$$

$$\mathbf{D}_H = \mathbf{Y}^\top \boldsymbol{\Omega}_H (\boldsymbol{\Omega}_H^\top \boldsymbol{\Omega}_H)^{-1}, \quad \mathbf{F}_H^Y = \mathbf{Y}_H.$$

At each step,  $d_h \mathbf{u}_h \mathbf{v}_h^\top$  is the best rank one approximation of  $\mathbf{M}_{h-1} := \mathbf{X}_{h-1}^\top \mathbf{Y}_{h-1}$  in the least squares sense and  $\mathbf{u}_h$  (resp.  $\mathbf{v}_h$ ) is given by the first left (resp. right) singular vector given by the SVD of  $\mathbf{M}_{h-1}$ , where  $d_h = \text{Cov}(\mathbf{X}_{h-1} \mathbf{u}_h, \mathbf{Y}_{h-1} \mathbf{v}_h)$  is the first (largest) singular value of this SVD.

We have the interesting recursion

$$\begin{aligned} \mathbf{M}_h &= \mathbf{X}_h^\top \mathbf{Y}_h \\ &= (\mathbf{X}_{h-1} - \boldsymbol{\xi}_h \mathbf{c}_h^\top)^\top (\mathbf{Y}_{h-1} - \boldsymbol{\omega}_h \mathbf{e}_h^\top) \\ &= \mathbf{X}_{h-1}^\top \mathbf{Y}_{h-1} - \mathbf{X}_{h-1}^\top \boldsymbol{\omega}_h \mathbf{e}_h^\top - \mathbf{c}_h \boldsymbol{\xi}_h^\top \mathbf{Y}_{h-1} \\ &\quad + \mathbf{c}_h \boldsymbol{\xi}_h^\top \boldsymbol{\omega}_h \mathbf{e}_h^\top \\ &= \mathbf{M}_{h-1} - \mathbf{M}_{h-1} \mathbf{v}_h \mathbf{e}_h^\top - \mathbf{c}_h \mathbf{u}_h^\top \mathbf{M}_{h-1} \\ &\quad + \mathbf{c}_h \mathbf{u}_h^\top \mathbf{M}_{h-1} \mathbf{v}_h \mathbf{e}_h^\top \\ &= (\mathbf{c}_h \mathbf{u}_h^\top - \mathbf{I}) \mathbf{M}_{h-1} (\mathbf{v}_h \mathbf{e}_h^\top - \mathbf{I}). \end{aligned}$$

Note that due to the constraints on the  $\boldsymbol{\xi}_j$ , we have that  $\boldsymbol{\Xi}_H^\top \boldsymbol{\Xi}_H$  is an invertible diagonal matrix and also that  $\boldsymbol{\xi}_h^\top = \boldsymbol{\xi}_h^\top \left( \prod_{j=h-1}^1 \mathcal{P}_{\boldsymbol{\xi}_j^\perp} \right) = \boldsymbol{\xi}_h^\top \mathcal{P}_{\boldsymbol{\Xi}_{\bullet h-1}^\perp}$ . This allows us

to write

$$\begin{aligned}
C_H^T &= (\Xi_H^T \Xi_H)^{-1} \Xi_H^T X \\
&= (\Xi_H^T \Xi_H)^{-1} \begin{bmatrix} \xi_1^T \\ \xi_2^T \mathcal{P}_{\Xi_{\bullet}^\perp} \\ \vdots \\ \xi_H^T \mathcal{P}_{\Xi_{H-1}^\perp} \end{bmatrix} X \\
&= (\Xi_H^T \Xi_H)^{-1} \begin{bmatrix} \xi_1^T X_0 \\ \xi_2^T X_1 \\ \vdots \\ \xi_H^T X_{H-1} \end{bmatrix} \\
&= \begin{bmatrix} (\xi_1^T \xi_1)^{-1} \xi_1^T X_0 \\ (\xi_2^T \xi_2)^{-1} \xi_2^T X_1 \\ \vdots \\ (\xi_H^T \xi_H)^{-1} \xi_H^T X_{H-1} \end{bmatrix}.
\end{aligned}$$

Similarly

$$D_H^T = \begin{bmatrix} (\omega_1^T \omega_1)^{-1} \omega_1^T Y_0 \\ (\omega_2^T \omega_2)^{-1} \omega_2^T Y_1 \\ \vdots \\ (\omega_H^T \omega_H)^{-1} \omega_H^T Y_{H-1} \end{bmatrix}.$$

These are the expressions given e.g., in [7, p. 10].

The linear combinations  $\xi_h = X_{h-1} u_h$ , and  $\omega_h = Y_{h-1} v_h$  are searched for recursively in the space spanned by the previous residuals. In what follows, we will consider how these linear combinations can be expressed in terms of the original variables. From Appendix A-F, we can write  $X_h = X A^{(h)}$  and  $Y_h = Y B^{(h)}$  with

$$A^{(h)} = \prod_{j=1}^h (I - u_j (\xi_j^T \xi_j)^{-1} \xi_j^T X)$$

and

$$B^{(h)} = \prod_{j=1}^h (I - v_j (\omega_j^T \omega_j)^{-1} \omega_j^T Y).$$

Defining  $\tilde{w}_h = \prod_{j=1}^{h-1} (I - u_j (\xi_j^T \xi_j)^{-1} \xi_j^T X) u_h$  and  $\tilde{z}_h = \prod_{j=1}^{h-1} (I - v_j (\omega_j^T \omega_j)^{-1} \omega_j^T Y) v_h$ , we have that  $\xi_h = X \tilde{w}_h$ , and  $\omega_h = Y \tilde{z}_h$ . These adjusted weights describe the effect of each of the original variables in constructing the scores  $\xi_h$  and  $\omega_h$ . To find what objective function these weights solve, we can use the argument from Appendix A-G to find:

$$u_h = \mathcal{P}_{\tilde{W}_{\bullet, h-1}^\perp} \tilde{w}_h, \quad v_h = \mathcal{P}_{\tilde{Z}_{\bullet, h-1}^\perp} \tilde{z}_h.$$

Substituting these equations into the objective function for the  $h$ -th set of PLS-W2A adjusted weights  $(\tilde{w}_h, \tilde{z}_h)$ , gives the equivalent optimisation problem

$$\arg\max_{\|\mathcal{P}_{\tilde{W}_{\bullet, h-1}^\perp} \tilde{w}\|_2 = \|\mathcal{P}_{\tilde{Z}_{\bullet, h-1}^\perp} \tilde{z}\|_2 = 1} \text{Cov}(X \tilde{w}, Y \tilde{z}).$$

(iii) The CCA objective function at step  $h$  is given by

$$(\tilde{w}_h, \tilde{z}_h) = \arg\max_{\tilde{w}, \tilde{z}} \text{Cor}(X \tilde{w}, Y \tilde{z}),$$

subject to the constraints

$$\text{Cov}(X \tilde{w}, X \tilde{w}_j) = \text{Cov}(Y \tilde{z}, Y \tilde{z}_j) = 0, \quad 1 \leq j < h.$$

Classical CCA relates  $X$  and  $Y$  by maximising the *correlation* between the scores (or canonical variates)  $\xi_h = X \tilde{w}_h$  and  $\omega_h = Y \tilde{z}_h$ , but without imposing a unit norm to the adjusted weights (or canonical) vectors  $\tilde{w}_h$  and  $\tilde{z}_h$ .

From the proof of (C2), and assuming that the  $X$  and  $Y$  sample covariance matrices are nonsingular (more on this later), an equivalent CCA objective function at step  $h$  is given by

$$\begin{cases} (u_h, v_h) = \arg\max_{\|u\|_2 = \|v\|_2 = 1} u^T M_0 v, \\ \tilde{w}_h = (X^T X)^{-1/2} u_h \text{ and } \tilde{z}_h = (Y^T Y)^{-1/2} v_h, \end{cases}$$

subject to the constraints  $u^T u_j = v^T v_j = 0$ ,  $1 \leq j < h$ , with  $M_0 := (X^T X)^{-1/2} X^T Y (Y^T Y)^{-1/2}$ .

Using (C2), the solution  $u_h$  (resp.  $v_h$ ) to this problem is the  $h$ -th column of the matrix  $U$  (resp.  $V$ ), obtained by applying (1) to  $M_0$ . Now, because of the imposed constraints on the  $u_h$  and  $v_h$ , we have

$$\begin{aligned}
u_h^T M_0 v_h &= u_h^T \left( M_0 - \sum_{l=1}^{h-1} \delta_h u_l v_l^T \right) v_h \\
&:= u_h^T M_{h-1} v_h.
\end{aligned}$$

It is thus possible to replace the above objective function with

$$\begin{cases} (u_h, v_h) = \arg\max_{\|u\|_2 = \|v\|_2 = 1} u^T M_{h-1} v, \\ \tilde{w}_h = (X^T X)^{-1/2} u_h \text{ and } \tilde{z}_h = (Y^T Y)^{-1/2} v_h, \end{cases}$$

where, thanks to the deflation property of the SVD, we have that  $\delta_h$ ,  $u_h$  and  $v_h$  are now obtained respectively (and successively) as the *first* singular value and *first* singular vectors of  $M_{h-1}$ . Iterations (deflations) are done using the relation  $M_h = M_{h-1} - \delta_h u_h v_h^T$ . Another approach is to define  $X_0 = X (X^T X)^{-1/2}$ ,  $Y_0 = Y (Y^T Y)^{-1/2}$ ,  $X_h = X_{h-1} (I - u_h u_h^T)$  and  $Y_h = Y_{h-1} (I - v_h v_h^T)$ . We have  $M_h = X_h^T Y_h$ . It follows that

$$\begin{aligned}
X_h^T Y_h &= (I - u_h u_h^T) X_{h-1}^T Y_{h-1} (I - v_h v_h^T) \\
&= \prod_{i=h}^1 (I - u_i u_i^T) X_0^T Y_0 \prod_{i=1}^h (I - v_i v_i^T) \\
&= (I - U_{\bullet, h} U_{\bullet, h}^T) U \Delta V^T (I - V_{\bullet, h} V_{\bullet, h}^T) \\
&= (U \Delta V^T - U_{\bullet, h} \Delta_h V_{\bullet, h}^T) (I - V_{\bullet, h} V_{\bullet, h}^T) \\
&= U \Delta V^T - U \Delta V^T V_{\bullet, h} V_{\bullet, h}^T - U_{\bullet, h} \Delta_h V_{\bullet, h}^T \\
&\quad + U_{\bullet, h} \Delta_h V_{\bullet, h}^T \\
&= U \Delta V^T - U_{\bullet, h} \Delta_h V_{\bullet, h}^T \\
&= M_0 - \sum_{i=1}^h \delta_i u_i v_i^T
\end{aligned}$$

where the product  $\prod_{i=1}^h (\mathbf{I} - \mathbf{u}_i \mathbf{u}_i^\top) = (\mathbf{I} - \mathbf{U}_{\bullet h} \mathbf{U}_{\bullet h}^\top)$  follows from [31, Theorem 7, p. 151], and where  $\mathcal{P}_{\mathbf{u}_h} = \mathbf{u}_h \mathbf{u}_h^\top$ .

It is thus possible to replace the objective function with

$$(\mathbf{u}_h, \mathbf{v}_h) = \underset{\|\mathbf{u}\|_2 = \|\mathbf{v}\|_2 = 1}{\operatorname{argmax}} \operatorname{Cov}(\mathbf{X}_{h-1} \mathbf{u}, \mathbf{Y}_{h-1} \mathbf{v}),$$

and to define the scores as  $\xi_h = \mathbf{X}_{h-1} \mathbf{u}_h$  and  $\omega_h = \mathbf{Y}_{h-1} \mathbf{v}_h$ .

Note that orthogonality of the scores holds due to the SVD properties:

$$\begin{aligned} \xi_j^\top \xi_h &= \tilde{\mathbf{w}}_j^\top \mathbf{X}^\top \mathbf{X} \tilde{\mathbf{w}}_h \\ &= \mathbf{u}_j^\top (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{X}^\top \mathbf{X} (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{u}_h \\ &= \mathbf{u}_j^\top \mathbf{u}_h \\ &= 0 \end{aligned}$$

for all  $j \neq h$ . Similarly, we find  $\omega_j^\top \omega_h = 0$  for  $j \neq h$ . We also have orthogonality between  $X$ - and  $Y$ -scores. For all  $j \neq h$ ,

$$\begin{aligned} \xi_j^\top \omega_h &= \tilde{\mathbf{w}}_j^\top \mathbf{X}^\top \mathbf{Y} \tilde{\mathbf{z}}_h \\ &= \mathbf{u}_j^\top (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{X}^\top \mathbf{Y} (\mathbf{Y}^\top \mathbf{Y})^{-1/2} \mathbf{v}_h \\ &= \mathbf{u}_j^\top \mathbf{M}_0 \mathbf{v}_h \\ &= \mathbf{u}_j^\top \left( \sum_{l=1}^r \delta_l \mathbf{u}_l \mathbf{v}_l^\top \right) \mathbf{v}_h \\ &= 0. \end{aligned}$$

Let  $\Xi_H = \mathbf{X} \tilde{\mathbf{W}}_H$  and  $\Omega_H = \mathbf{Y} \tilde{\mathbf{Z}}_H$ , where  $\tilde{\mathbf{W}}_H = (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{U}_{\bullet H}$  and  $\tilde{\mathbf{Z}}_H = (\mathbf{Y}^\top \mathbf{Y})^{-1/2} \mathbf{V}_{\bullet H}$ . Since we have assumed that  $\mathbf{X}^\top \mathbf{X}$  is invertible, we have

$$\begin{aligned} \Xi_H^\top \Xi_H &= \mathbf{U}_{\bullet H}^\top (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{X}^\top \mathbf{X} (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{U}_{\bullet H} \\ &= \mathbf{U}_{\bullet H}^\top \mathbf{U}_{\bullet H} \\ &= \mathbf{I}_H. \end{aligned}$$

Similarly,  $\Omega_H^\top \Omega_H = \mathbf{I}_H$ . We have

$$\begin{aligned} \mathbf{X} &= \mathcal{P}_{\Xi_H} \mathbf{X} + \mathcal{P}_{\Xi_H^\perp} \mathbf{X} \\ &= \Xi_H (\Xi_H^\top \Xi_H)^{-1} \Xi_H^\top \mathbf{X} + \mathcal{P}_{\Xi_H^\perp} \mathbf{X} \\ &= \Xi_H \Xi_H^\top \mathbf{X} + \mathcal{P}_{\Xi_H^\perp} \mathbf{X}. \end{aligned}$$

A similar expression holds for  $\mathbf{Y}$ . Thus the elements of the decomposition model (3) are

$$\begin{aligned} \mathbf{C}_H &= \mathbf{X}^\top \Xi_H, \quad \mathbf{F}_H^X = \mathcal{P}_{\Xi_H^\perp} \mathbf{X}; \\ \mathbf{D}_H &= \mathbf{Y}^\top \Omega_H, \quad \mathbf{F}_H^Y = \mathcal{P}_{\Omega_H^\perp} \mathbf{Y}. \end{aligned}$$

It has been suggested [32, p. 287], [33, p. 75] to use generalised inverses (e.g., Moore-Penrose) to deal with the singular case, and use the objective function at step  $h$

$$\left\{ \begin{aligned} (\mathbf{u}_h, \mathbf{v}_h) &= \underset{\|\mathbf{u}\|_2 = \|\mathbf{v}\|_2 = 1}{\operatorname{argmax}} \mathbf{u}^\top \mathbf{M}_0 \mathbf{v}, \\ \tilde{\mathbf{w}}_h &= (\mathbf{X}^\top \mathbf{X})^{+1/2} \mathbf{u}_h \quad \text{and} \quad \tilde{\mathbf{z}}_h = (\mathbf{Y}^\top \mathbf{Y})^{+1/2} \mathbf{v}_h, \end{aligned} \right.$$

where  $\mathbf{M}_0 = (\mathbf{X}^\top \mathbf{X})^{+1/2} \mathbf{X}^\top \mathbf{Y} (\mathbf{Y}^\top \mathbf{Y})^{+1/2}$ .

This being said, this approach can produce a meaningless solution, with correlations trivially equal to one. Indeed, there exists infinitely many other generalised inverses. They might lead to other weights and scores, still with the same optimal correlation between scores. Moreover, in this case, a small change in the data can lead to large changes in the weights and scores [7, pp. 26–27]. In other words, overfitting would occur.

An alternative for the case of singular matrices is to perform regularisation on the sample covariance matrices. The regularised solution trades off bias for a lower variance solution. Regularisation was first introduced to the CCA method by [34] and later refined by [35]. This method is known as regularised CCA (rCCA) or canonical ridge analysis and is closely related to Tikhonov's regularisation (or ridge regression) for the solution of systems of linear equations. This regularisation is imposed by replacing the matrices  $\mathbf{X}^\top \mathbf{X}$  and  $\mathbf{Y}^\top \mathbf{Y}$  with  $\mathbf{X}^\top \mathbf{X} + \lambda_x \mathbf{I}_p$  and  $\mathbf{Y}^\top \mathbf{Y} + \lambda_y \mathbf{I}_q$  respectively in the optimisation criterion. The regularisation parameters  $\lambda_x$  and  $\lambda_y$  should be nonnegative and if they are nonzero, then the regularised covariance matrices will be nonsingular. With a slightly different application of the regularisation parameters we could use:

$$\begin{aligned} (1 - \lambda_x^*) \mathbf{X}^\top \mathbf{X} + \lambda_x^* \mathbf{I}_p \\ (1 - \lambda_y^*) \mathbf{Y}^\top \mathbf{Y} + \lambda_y^* \mathbf{I}_q, \end{aligned}$$

with  $0 \leq \lambda_x^*, \lambda_y^* \leq 1$ . The objective function in this case changes to [12, p. 38]:

$$\left\{ \begin{aligned} (\mathbf{u}_h, \mathbf{v}_h) &= \underset{\|\mathbf{u}\|_2 = \|\mathbf{v}\|_2 = 1}{\operatorname{argmax}} \mathbf{u}^\top \mathbf{M}_0 \mathbf{v}, \\ \tilde{\mathbf{w}}_h &= ((1 - \lambda_x^*) \mathbf{X}^\top \mathbf{X} + \lambda_x^* \mathbf{I}_p)^{-1/2} \mathbf{u}_h \\ \tilde{\mathbf{z}}_h &= ((1 - \lambda_y^*) \mathbf{Y}^\top \mathbf{Y} + \lambda_y^* \mathbf{I}_q)^{-1/2} \mathbf{v}_h, \end{aligned} \right.$$

where  $\mathbf{M}_0$  is defined as

$$((1 - \lambda_x^*) \mathbf{X}^\top \mathbf{X} + \lambda_x^* \mathbf{I}_p)^{-1/2} \mathbf{X}^\top \mathbf{Y} ((1 - \lambda_y^*) \mathbf{Y}^\top \mathbf{Y} + \lambda_y^* \mathbf{I}_q)^{-1/2}.$$

Note that ordinary CCA is obtained at  $\lambda_x^* = \lambda_y^* = 0$ , and PLS-SVD is obtained with  $\lambda_x^* = \lambda_y^* = 1$ . Other approaches exist; see e.g., [23, eq. (13)].

(iv) PLS-R (also called PLS1 if  $q = 1$  or PLS2 if  $q > 1$ ) is a regression technique that predicts one set of data from another, hence termed asymmetric, while describing their common structure. It finds latent variables (also called component scores) that model  $\mathbf{X}$  and simultaneously predict  $\mathbf{Y}$ . While several algorithms have been developed to solve this problem, we focus on the two most well known variants. The first, is an extension of the Nonlinear estimation by Iterative Partial Least Squares (NIPALS), initially proposed by H. Wold [1] as an alternative algorithm for implementing Principal Component Analysis, and modified by [36] to obtain a regularized component based regression tool. The second, is the Statistically Inspired Modification of PLS (SIMPLS) [37]. We now give some details about outputs of these two algorithms. Other PLS regression algorithms can be found in [38], and see also [39] for a

numerical comparison.

The  $h$ -th set of PLS regression weights  $(\mathbf{u}_h, \mathbf{v}_h)$  given by NIPALS solve the optimisation problem [40, eq. (5)]

$$(\mathbf{u}_h, \mathbf{v}_h) = \underset{\|\mathbf{u}\|_2=\|\mathbf{v}\|_2=1}{\operatorname{argmax}} \operatorname{Cov}(\mathbf{X}_{h-1}\mathbf{u}, \mathbf{Y}_{h-1}\mathbf{v}),$$

where the deflated matrices are defined by  $\mathbf{X}_0 := \mathbf{X}$ ,  $\mathbf{Y}_0 := \mathbf{Y}$ ,

$$\mathbf{X}_h = \mathbf{X}_{h-1} - \xi_h(\xi_h^\top \xi_h)^{-1} \xi_h^\top \mathbf{X}_{h-1} = \mathcal{P}_{\xi_h^\perp} \mathbf{X}_{h-1},$$

with  $\xi_h = \mathbf{X}_{h-1}\mathbf{u}_h$ , and

$$\mathbf{Y}_h = \mathbf{Y}_{h-1} - \xi_h(\xi_h^\top \xi_h)^{-1} \xi_h^\top \mathbf{Y}_{h-1} = \mathcal{P}_{\xi_h^\perp} \mathbf{Y}_{h-1}.$$

Classical PLS-R searches for successive  $X$ -score vectors  $\xi_h$  (stored in the matrix  $\Xi_H$ ) that are orthogonal to the previous ones by construction ( $\xi_h = \mathcal{P}_{\Xi_{h-1}^\perp} \mathbf{X}\mathbf{u}_h$ ) and  $Y$ -score vectors  $\omega_h$  (defined below and stored in the matrix  $\Omega_H$ ). The first pair  $(\xi_1, \omega_1)$  of  $X$ - and  $Y$ -score vectors is the one with maximal covariance. The next pairs are searched for using successively deflated versions of  $\mathbf{X}$  and of  $\mathbf{Y}$ . We thus remove the information contained in the previous  $X$ -scores here. From (C1), the solution  $\mathbf{u}_h$  (resp.  $\mathbf{v}_h$ ) is the *first* left (resp. right) singular vector of  $\mathbf{M}_{h-1} := \mathbf{X}_{h-1}^\top \mathbf{Y}_{h-1}$ .

We have the interesting recursion

$$\begin{aligned} \mathbf{M}_h &= \mathbf{X}_h^\top \mathbf{Y}_h \\ &= (\mathbf{X}_{h-1} - \xi_h c_h^\top)^\top (\mathbf{Y}_{h-1} - \xi_h d_h^\top) \\ &= \mathbf{X}_{h-1}^\top \mathbf{Y}_{h-1} - \mathbf{X}_{h-1}^\top \xi_h d_h^\top - c_h \xi_h^\top \mathbf{Y}_{h-1} \\ &\quad + c_h \xi_h^\top \xi_h d_h^\top \\ &= \mathbf{M}_{h-1} - \mathbf{X}_{h-1}^\top \mathbf{X}_{h-1} \mathbf{u}_h d_h^\top - c_h \mathbf{u}_h^\top \mathbf{M}_{h-1} \\ &\quad + c_h \mathbf{u}_h^\top \mathbf{X}_{h-1}^\top \mathbf{X}_{h-1} \mathbf{u}_h d_h^\top \\ &= (\mathbf{I} - c_h \mathbf{u}_h^\top) \mathbf{M}_{h-1} - (\mathbf{I} - c_h \mathbf{u}_h^\top) \mathbf{N}_{h-1} \mathbf{u}_h d_h^\top \\ &= (\mathbf{I} - c_h \mathbf{u}_h^\top) (\mathbf{M}_{h-1} - \mathbf{N}_{h-1} \mathbf{u}_h d_h^\top), \end{aligned}$$

where  $\mathbf{N}_0 = \mathbf{X}_0^\top \mathbf{X}_0$  and

$$\mathbf{N}_h = (c_h \mathbf{u}_h^\top - \mathbf{I}) \mathbf{N}_{h-1} (\mathbf{u}_h c_h^\top - \mathbf{I}).$$

Note that the original NIPALS algorithm computes the above quantities using an iterative procedure designed to compute eigenvectors (but see the relation between eigenvectors and singular vectors in Appendix A-C). Two versions are found in the literature, whether  $\mathbf{v}_h$  is scaled [41] or not [36], [42]. At the end of both algorithms, the fitted values  $\hat{\mathbf{Y}}_H$  are computed [43, Equ. (20)]

$$\hat{\mathbf{Y}}_H = \mathcal{P}_{\Xi_H} \mathbf{Y} = \Xi_H (\Xi_H^\top \Xi_H)^{-1} \Xi_H^\top \mathbf{Y}.$$

This is described in Appendix A-D.

Let  $\alpha_h = \|\xi_h\|^2 / \|\mathbf{Y}_{h-1}^\top \xi_h\|$ . Now, define  $p_h = \alpha_h^{-1}$  in the scaled case and  $p_h = 1$  otherwise. The  $Y$ -score vectors are defined as  $\omega_h = p_h \alpha_h \mathbf{Y}_{h-1} \mathbf{v}_h$ ,  $h = 1, \dots, H$ . In addition to the usual decomposition equations (3) that will be explicitied below, the PLS regression algorithm

includes an additional ‘‘inner relationship’’ which relates the  $Y$ -scores  $\Omega_{\bullet h}$  to the  $X$ -scores  $\Xi_{\bullet h}$  explicitly:

$$\Omega_{\bullet h} = \Xi_{\bullet h} \mathbf{P}_h + \mathbf{R}_{\bullet h}, \quad (4)$$

where  $\Omega_{\bullet h} = (\omega_j)_{1 \leq j \leq h}$ ,  $\mathbf{P}_h = \operatorname{diag}(p_j)_{1 \leq j \leq h}$  and where  $\mathbf{R}_{\bullet h}$  is a matrix of residuals. Note that in the unscaled case,  $\mathbf{P}_h = \mathbf{I}$ . Proof is provided in Appendix A-E.

The decomposition model (3) is given by (see Appendix A-F for proof):

$$\mathbf{X} = \mathcal{P}_{\Xi_H} \mathbf{X} + \mathcal{P}_{\Xi_H^\perp} \mathbf{X} = \Xi_H \mathbf{C}_H^\top + \mathbf{X}_H,$$

where  $\mathbf{C}_H = \mathbf{X}^\top \Xi_H (\Xi_H^\top \Xi_H)^{-1}$  and where the matrix of residuals is  $\mathbf{F}_H^X = \mathbf{X}_H = \mathcal{P}_{\Xi_H^\perp} \mathbf{X}$ . We have

$$\begin{aligned} \mathbf{Y} &= \Omega_H \mathbf{D}_H^\top + \mathbf{F}_H^Y \\ &= (\Xi_H \mathbf{P}_H + \mathbf{R}_H) \mathbf{D}_H^\top + \mathbf{F}_H^Y \\ &= \Xi_H \mathbf{P}_H \mathbf{D}_H^\top + (\mathbf{R}_H \mathbf{D}_H^\top + \mathbf{F}_H^Y) \\ &= \mathbf{X} \tilde{\mathbf{W}}_H \mathbf{P}_H \mathbf{D}_H^\top + (\mathbf{R}_H \mathbf{D}_H^\top + \mathbf{F}_H^Y) \\ &= \mathbf{X} \hat{\mathbf{B}}_{PLS} + \mathbf{E}_H^Y, \end{aligned}$$

where  $\mathbf{D}_H = [\mathbf{v}_1, \dots, \mathbf{v}_H]$ ,  $\hat{\mathbf{B}}_{PLS} = \mathbf{U}_{\bullet H} (\mathbf{C}_H^\top \mathbf{U}_{\bullet H})^{-1} \mathbf{P}_H \mathbf{D}_H^\top := \tilde{\mathbf{W}}_H \mathbf{P}_H \mathbf{D}_H^\top$ , and where the matrices of residuals are  $\mathbf{F}_H^Y = \Xi_H \mathbf{G}_H^\top - \Omega_H \mathbf{D}_H^\top + \mathbf{Y}_H$  and  $\mathbf{E}_H^Y = \mathbf{R}_H \mathbf{D}_H^\top + \mathbf{F}_H^Y$ . The  $h$ -th row of  $\mathbf{C}_H^\top$  and  $\mathbf{G}_H^\top$  are respectively  $(\xi_h^\top \xi_h)^{-1} \xi_h^\top \mathbf{X}_{h-1}$  and  $(\xi_h^\top \xi_h)^{-1} \xi_h^\top \mathbf{Y}_{h-1}$ .

**Remark 1.** For univariate regression, the objective function can be restated as follows [44]:

$$\mathbf{u}_h = \underset{\|\mathbf{u}\|_2=1}{\operatorname{argmax}} \operatorname{Cor}^2(\mathbf{X}_{h-1}\mathbf{u}, \mathbf{Y}_{h-1}) \operatorname{Var}(\mathbf{X}_{h-1}\mathbf{u}),$$

where  $v = 1$  since the response  $\mathbf{Y}_{h-1} : n \times 1$  is univariate, and where we have used the relationship

$$\begin{aligned} \operatorname{Cov}^2(\mathbf{X}_{h-1}\mathbf{u}, \mathbf{Y}_{h-1}) &= \\ \operatorname{Var}(\mathbf{X}_{h-1}\mathbf{u}) \operatorname{Cor}^2(\mathbf{X}_{h-1}\mathbf{u}, \mathbf{Y}_{h-1}) \operatorname{Var}(\mathbf{Y}_{h-1}). \end{aligned}$$

This formulation shows that PLS seeks directions that relate  $\mathbf{X}$  and  $\mathbf{Y}$  by maximising the correlation, and capture the most variable directions in the  $X$ -space.

We now present equivalent objective functions that one can encounter in the literature. Since the optimal solution  $\mathbf{v}_h$  to the objective problem should be proportional to  $\mathbf{M}_{h-1}^\top \mathbf{u}_h$  (see Proof of (C1) in the Appendix A-A), the optimisation problem is equivalent to [45, eq. (13)–(14)]

$$\begin{cases} \mathbf{u}_h = \underset{\|\mathbf{u}\|_2=1}{\operatorname{argmax}} (\mathbf{u}^\top \mathbf{M}_{h-1} \mathbf{M}_{h-1}^\top \mathbf{u}) \\ \tilde{\mathbf{v}}_h = \mathbf{M}_{h-1}^\top \mathbf{u}_h ; \text{ norm } \tilde{\mathbf{v}}_h, \end{cases}$$

whose solution can be obtained using the so-called PLS2 algorithm [40]. We note that only one of  $\mathbf{X}$  or  $\mathbf{Y}$  needs to be deflated, [40] because:

$$\begin{aligned} \mathbf{M}_h &= \mathbf{X}_h^\top \mathbf{Y}_h \\ &= \mathbf{X}^\top \mathcal{P}_{\Xi_{\bullet,h}}^\top \mathcal{P}_{\Xi_{\bullet,h}} \mathbf{Y} \\ &= \mathbf{X}^\top \mathcal{P}_{\Xi_{\bullet,h}}^\top \mathbf{Y}, \end{aligned}$$

which is equal to  $\mathbf{X}_h^\top \mathbf{Y}$  (or  $\mathbf{X}^\top \mathbf{Y}_h$ ). Thus the previous optimisation problem can be written as [40, eq. (7)]:

$$\begin{cases} \mathbf{u}_h = \underset{\|\mathbf{u}\|_2=1}{\operatorname{argmax}} (\mathbf{u}^\top \mathbf{X}_{h-1}^\top \mathbf{Y} \mathbf{Y}^\top \mathbf{X}_{h-1} \mathbf{u}) \\ \tilde{\mathbf{v}}_h = \mathbf{Y}^\top \mathbf{X}_{h-1} \mathbf{u}_h; \text{ norm } \tilde{\mathbf{v}}_h. \end{cases}$$

Similar to PLS-W2A, the linear combinations  $\xi_h = \mathbf{X}_{h-1} \mathbf{u}_h$  are searched for recursively through the successive residuals. We now consider the construction of the scores in terms of the original variables  $\xi_h = \mathbf{X} \tilde{\mathbf{w}}_h$ . From Appendix A-G, we have

$$\mathbf{u}_h = \mathcal{P}_{\tilde{\mathbf{W}}_{\bullet,h-1}^\perp} \tilde{\mathbf{w}}_h.$$

The above optimisation problem is thus equivalent to solving [25, eq. (2)]

$$\begin{cases} \tilde{\mathbf{w}}_h = \underset{\|\mathcal{P}_{\tilde{\mathbf{W}}_{\bullet,h-1}^\perp} \tilde{\mathbf{w}}\|_2=1}{\operatorname{argmax}} (\tilde{\mathbf{w}}^\top \mathbf{X}^\top \mathbf{Y} \mathbf{Y}^\top \mathbf{X} \tilde{\mathbf{w}}) \\ \tilde{\mathbf{v}}_h = \mathbf{Y}^\top \mathbf{X} \tilde{\mathbf{w}}_h; \text{ norm } \tilde{\mathbf{v}}_h, \end{cases} \quad (5)$$

(without deflations), this is the so-called “PLS2” objective function.

The second most commonly used PLSR algorithm, called SIMPLS [37], calculates the PLS latent components directly as linear combinations of the original variables. The objective function to optimise is [25, eq. (3)]

$$\mathbf{w}_h = \underset{\|\mathbf{w}\|_2=1}{\operatorname{argmax}} (\mathbf{w}^\top \mathbf{X}^\top \mathbf{Y} \mathbf{Y}^\top \mathbf{X} \mathbf{w}),$$

(without deflations) subject to the constraints

$$\operatorname{Cov}(\mathbf{X} \mathbf{w}, \mathbf{X} \mathbf{w}_j) = 0, \quad 1 \leq j < h,$$

from which we compute

$$\mathbf{u}_h = \mathbf{w}_h, \quad \mathbf{v}_h = \mathbf{Y}^\top \mathbf{X} \mathbf{w}_h / \|\mathbf{Y}^\top \mathbf{X} \mathbf{w}_h\|_2.$$

It is important to note that both algorithms have the same objective function but different constraints and thus yield different sets of direction vectors. The solution  $\mathbf{w}_h$  to SIMPLS is the first left singular vector of  $\mathbf{M}_{h-1} := \mathcal{P}_{\mathbf{C}_{\bullet,h-1}^\perp} \mathbf{X}^\top \mathbf{Y}$  [43, p. 322].

**Remark 2.** Another equivalent objective function for SIMPLS is [14]

$$(\mathbf{w}_h, \mathbf{v}_h) = \underset{\|\mathbf{w}\|_2=\|\mathbf{v}\|_2=1}{\operatorname{argmax}} (\mathbf{w}^\top \mathbf{X}^\top \mathbf{Y} \mathbf{v})$$

subject to the constraints

$$\operatorname{Cov}(\mathbf{X} \mathbf{w}, \mathbf{X} \mathbf{w}_j) = 0, \quad 1 \leq j < h.$$

The decomposition model for SIMPLS is identical to the decomposition of PLS2, the only difference being in how the weights  $\mathbf{u}_h$  are calculated. In both models we have  $\Xi_{\bullet,h} = \mathbf{X} \mathbf{W}_{\bullet,h}$  (or  $\Xi_{\bullet,h} = \mathbf{X} \tilde{\mathbf{W}}_{\bullet,h}$ ), but the different constraints on the adjusted weights  $\mathbf{w}_h$  (or  $\tilde{\mathbf{w}}_h$ ) give different score vectors;  $\|\mathbf{w}_h\|_2 = 1$  versus  $\|\mathcal{P}_{\tilde{\mathbf{W}}_{\bullet,h-1}^\perp} \tilde{\mathbf{w}}_h\|_2 = 1$ . The predictions for both models are generated via  $\hat{\mathbf{Y}}_h = \mathcal{P}_{\Xi_{\bullet,h}} \mathbf{Y}$  so they will produce different predictions.

**Remark 3.** Another closely related (to SIMPLS) algorithm is the PLS simple iteration algorithm [46]. It has exactly the same objective function (and thus gives the same results) but differs in the way the matrices  $\mathcal{P}_{\mathbf{C}_{\bullet,h-1}}$  are computed. For the latter, the recursion formula  $\mathbf{M}_h = \mathbf{M}_{h-1} - \mathcal{P}_{\mathbf{M}_{h-1} \mathbf{X}^\top \mathbf{X} \mathbf{w}_h}$  is used.

### III. PENALIZED PLS

All of the previous PLS methods can be written in terms of a single optimisation problem coupled with an appropriate deflation to ensure the appropriate orthogonal constraints. In this section we introduce the framework for penalised partial least squares in the unified PLS methodology. Several penalisations are then considered and presented in a unified algorithm that can preform all four PLS methods, and their regularised versions.

#### A. Finding the PLS weights

The  $h$ -th pair of penalised PLS weight vectors are given by the algorithm in Section C, where  $P_{\lambda_1}$  and  $P_{\lambda_2}$  are convex penalty functions with tuning parameters  $\lambda_1$  and  $\lambda_2$ , and the matrix  $\mathbf{M}_{h-1}$  is defined in the appropriate subsection of Section E. The resulting objective function solved at each iteration is convex in  $\tilde{\mathbf{u}}$  (with fixed  $\tilde{\mathbf{v}}$ ) and convex in  $\tilde{\mathbf{v}}$  (for fixed  $\tilde{\mathbf{u}}$ ). For a fixed unit norm  $\tilde{\mathbf{v}}$ , using the SVD connection, the optimisation is

$$\tilde{\mathbf{u}}_h = \underset{\tilde{\mathbf{u}}}{\operatorname{argmin}} \left\{ \|\mathbf{M}_{h-1} - \tilde{\mathbf{u}} \tilde{\mathbf{v}}^\top\|_F^2 + P_{\lambda_1}(\tilde{\mathbf{u}}) \right\}, \quad (6)$$

and we set  $\mathbf{u}_h = \tilde{\mathbf{u}}_h / \|\tilde{\mathbf{u}}_h\|_2$  if  $\|\tilde{\mathbf{u}}_h\|_2 > 0$  and  $\mathbf{u}_h = \mathbf{0}_p$  otherwise. Similarly, for a fixed unit norm  $\tilde{\mathbf{u}}$  we solve the optimisation problem

$$\tilde{\mathbf{v}}_h = \underset{\tilde{\mathbf{v}}}{\operatorname{argmin}} \left\{ \|\mathbf{M}_{h-1}^\top - \tilde{\mathbf{v}} \tilde{\mathbf{u}}^\top\|_F^2 + P_{\lambda_2}(\tilde{\mathbf{v}}) \right\}, \quad (7)$$

and set  $\mathbf{v}_h = \tilde{\mathbf{v}}_h / \|\tilde{\mathbf{v}}_h\|_2$  if  $\|\tilde{\mathbf{v}}_h\|_2 > 0$  and  $\mathbf{v}_h = \mathbf{0}_q$  otherwise. For certain penalty functions, the convergence of this algorithm has been studied [23].

#### B. Deflation and the PLS weights

Computing the penalised versions of the four different PLS methods is achieved by alternating between two subtasks: solving (6) and (7) for the weights, and matrix deflation. Without the penalties  $P_{\lambda_1}$  and  $P_{\lambda_2}$ , the matrix deflation enforces certain orthogonality constraints for each of the four standard PLS methods. However, with either penalty  $P_{\lambda_1}$  or  $P_{\lambda_2}$ , these deflations do not ensure any orthogonal constraints.



Although, these constraints are lost, Witten et al. [23], state that it is not clear that orthogonality is desirable as it may be at odds with sparsity. That is, enforcing the additional orthogonality constraints may result in less sparse solutions. Similar to [21], [23] we use the standard deflation methods in our implementation of the penalised PLS methods. Alternative matrix deflations have been proposed for sparse PCA [47]. However, these methods have not been extended in the general penalised PLS framework.

Another key observation is that for the NIPALS PLS regression, PLS-W2A, and CCA the scores were defined in terms of the deflated matrices  $\xi_h = \mathbf{X}_{h-1}\mathbf{u}_h$  and  $\omega_h = \mathbf{Y}_{h-1}\mathbf{v}_h$ . Consequently the sparse estimates given by solving (6) and (7) perform variable selection of the deflated matrices. Thus the latent components formed using these methods have the interpretation given by Remark 4. In our implementation, we also calculate the adjusted weights  $\mathbf{w}_h$  and  $\mathbf{z}_h$  (or  $\tilde{\mathbf{w}}_h$  and  $\tilde{\mathbf{z}}_h$ ), where  $\xi_h = \mathbf{X}\mathbf{w}_h$  and  $\omega_h = \mathbf{Y}\mathbf{z}_h$ . These weights allow for direct interpretation of the selected variables in the PLS model. Note that although  $\mathbf{w}_h$  and  $\mathbf{z}_h$  allow for direct interpretation of the selected variables, the sparsity is enforced on  $\mathbf{u}_h$  and  $\mathbf{v}_h$ . So if  $\mathbf{u}_h$  and  $\mathbf{v}_h$  are sparse, this does not necessarily mean that the adjusted weights  $\mathbf{w}_h$  and  $\mathbf{z}_h$  will be sparse.

**Remark 4.** *It is important to understand how to interpret the results of such an analysis. The first latent variable  $\xi_1 = \mathbf{X}\mathbf{u}_1$  is built as a sparse linear combination (with weights in  $\mathbf{u}_1$ ) of the original variables. The next latent variable  $\xi_2 = \mathcal{P}_{\xi_1^\perp}\mathbf{X}\mathbf{u}_2$  is the part of the sparse linear combination (with weights in  $\mathbf{u}_2$ ) of the original variables that has not been already explained by the first latent variable. And more generally, the  $h$ -th latent variable is built as a sparse linear combination of the original variables, from which we extract (by projection) the information not already brought by the previous latent variables.*

We note that an alternative SIMPLS formulation for the penalised PLS methods was proposed in a regression setting by [48]. In the SIMPLS method the weights are directly interpreted in terms of the original variables, so  $\mathbf{w}_h = \mathbf{u}_h$  and  $\mathbf{z}_h = \mathbf{v}_h$ . Although this method allows for direct penalisation of the weights, the orthogonality conditions still do not hold. We have incorporated this method and a similar variant for PLS-W2A in our package `bigsgPLS` to allow for direct penalisation of the weights.

### C. The penalised PLS methods

Computationally, the PLS method is an efficient approach to sparse latent variable modelling. The main computational cost is in solving for the PLS weights as described in equations (6) and (7). The cost of solving for these weights is penalty specific but can be minimal in a number of useful applications. We detail a few examples where these equations have been solved analytically, and provide an algorithm that treats the penalised versions of the four PLS cases (i) – (iv).

1) *Sparse PLS*: The (original) sparse PLS version sPLS [21] (see also [25]) considers the following penalty functions

$$P_{\lambda_1}(\tilde{\mathbf{u}}) = \sum_{i=1}^p 2\lambda_1|\tilde{u}_i| \quad \text{and} \quad P_{\lambda_2}(\tilde{\mathbf{v}}) = \sum_{j=1}^q 2\lambda_2|\tilde{v}_j|. \quad (8)$$

These penalties induce the desired sparsity of the weight vectors  $\mathbf{u}_h = \tilde{\mathbf{u}}_h/\|\tilde{\mathbf{u}}_h\|_2$  and  $\mathbf{v}_h = \tilde{\mathbf{v}}_h/\|\tilde{\mathbf{v}}_h\|_2$ , thanks to the well known properties of the  $\ell_1$ -norm or Lasso penalty [49]. The closed form solution for this problem is (see Appendix A-H for proof):

$$\begin{aligned} \tilde{\mathbf{u}} &= g^{\text{soft}}(\mathbf{M}\mathbf{v}, \lambda_1), \\ \tilde{\mathbf{v}} &= g^{\text{soft}}(\mathbf{M}^\top\mathbf{u}, \lambda_2). \end{aligned} \quad (9)$$

where  $g^{\text{soft}}(\cdot, \lambda_1)$  is the soft thresholding function, with the understanding that the function is applied componentwise. To unify these results with the ones to come, we introduce the sparsifier functions  $S_u$  and  $S_v$  to denote analytical functions that provide the solution for the weights. The sparsifiers are functions of the data  $\mathbf{M}$ , the fixed weight  $\mathbf{u}$  (or  $\mathbf{v}$ ) and additional penalty specific parameters  $\theta_u$  (or  $\theta_v$ ). So for sparse PLS we have,

$$\begin{aligned} \tilde{\mathbf{u}}_h &= S_u(\mathbf{v}; \mathbf{M}, \theta_u) = g^{\text{soft}}(\mathbf{M}\mathbf{v}, \lambda_1), \\ \tilde{\mathbf{v}}_h &= S_v(\mathbf{u}; \mathbf{M}, \theta_v) = g^{\text{soft}}(\mathbf{M}^\top\mathbf{u}, \lambda_2), \end{aligned} \quad (10)$$

where  $\theta_u = \lambda_1$  and  $\theta_v = \lambda_2$ .

2) *Group PLS*: There are many statistical problems in which the data has a natural grouping structure. In these problems, it is preferable to estimate all coefficients within a group to be zero or nonzero simultaneously. A leading example is in gene expression data, where genes within the same gene pathway have a similar biological function. Selecting a group amounts to selecting a pathway. Variables can be grouped for other reasons. For example, when we have categorical covariates in our data. This data is coded by their factor levels using dummy variables, and we desire selection or exclusion of this group of dummy variables.

Let us consider a situation where both matrices  $\mathbf{X}$  and  $\mathbf{Y}$  can be divided respectively into  $K$  and  $L$  sub-matrices (i.e., groups)  $\mathbf{X}^{(k)} : n \times p_k$  and  $\mathbf{Y}^{(l)} : n \times q_l$ , where  $p_k$  (resp.  $q_l$ ) is the number of covariates in group  $k$  (resp.  $l$ ). The aim is to select only a few groups of  $\mathbf{X}$  which are related to a few groups of  $\mathbf{Y}$ . We define  $\mathbf{M}^{(k,\cdot)} = \mathbf{X}^{(k)\top}\mathbf{Y}$  and  $\mathbf{M}^{(\cdot,l)} = \mathbf{Y}^{(l)\top}\mathbf{X}$ .

Group selection is accomplished using the group lasso penalties [50] in the optimisation problems (6) and (7):

$$\begin{aligned} P_{\lambda_1}(\tilde{\mathbf{u}}) &= \lambda_1 \sum_{k=1}^K \sqrt{p_k} \|\tilde{\mathbf{u}}^{(k)}\|_2; \\ P_{\lambda_2}(\tilde{\mathbf{v}}) &= \lambda_2 \sum_{l=1}^L \sqrt{q_l} \|\tilde{\mathbf{v}}^{(l)}\|_2, \end{aligned} \quad (11)$$

where  $\tilde{\mathbf{u}}^{(k)}$  and  $\tilde{\mathbf{v}}^{(l)}$  are the sub vectors of the (unscaled) weights  $\tilde{\mathbf{u}}$  and  $\tilde{\mathbf{v}}$  corresponding to the variables in group  $k$  of  $\mathbf{X}$  and group  $l$  of  $\mathbf{Y}$  respectively. This penalty is a group generalisation of the Lasso penalty. Depending on the tuning

parameter  $\lambda_1 \geq 0$  (or  $\lambda_2 \geq 0$ ), the entire weight subvector  $\tilde{\mathbf{u}}^{(k)}$  (or  $\tilde{\mathbf{v}}^{(l)}$ ) will be zero, or nonzero together.

The closed form solution for the group PLS method for the  $k$ -th subvector of  $\tilde{\mathbf{u}}$  is given by

$$S_u^{(k)}(\tilde{\mathbf{v}}; \mathbf{M}, \boldsymbol{\theta}_u) = \left(1 - \frac{\lambda_1}{2} \frac{\sqrt{p_k}}{\|\mathbf{M}^{(k,\cdot)} \tilde{\mathbf{v}}\|_2}\right)_+ \mathbf{M}^{(k,\cdot)} \tilde{\mathbf{v}}, \quad (12)$$

so  $\tilde{\mathbf{u}}^{(k)} = S_u^{(k)}(\tilde{\mathbf{v}}; \mathbf{M}, \boldsymbol{\theta}_u)$ . Similarly, the closed form solution for the  $l$ -th subvector of  $\tilde{\mathbf{v}}$  is

$$S_v^{(l)}(\tilde{\mathbf{u}}; \mathbf{M}, \boldsymbol{\theta}_v) = \left(1 - \frac{\lambda_2}{2} \frac{\sqrt{q_l}}{\|\mathbf{M}^{(\cdot,l)} \tilde{\mathbf{u}}\|_2}\right)_+ \mathbf{M}^{(\cdot,l)} \tilde{\mathbf{u}}, \quad (13)$$

so  $\tilde{\mathbf{v}}^{(l)} = S_v^{(l)}(\tilde{\mathbf{u}}; \mathbf{M}, \boldsymbol{\theta}_v)$ . The sparsifyer functions are applied groupwise

$$\begin{aligned} \tilde{\mathbf{u}} &= S_u(\tilde{\mathbf{v}}; \mathbf{M}, \boldsymbol{\theta}_u) = \left(S_u^{(1)}(\tilde{\mathbf{v}}; \mathbf{M}, \boldsymbol{\theta}_u), \dots, S_u^{(K)}(\tilde{\mathbf{v}}; \mathbf{M}, \boldsymbol{\theta}_u)\right) \\ \tilde{\mathbf{v}} &= S_v(\tilde{\mathbf{u}}; \mathbf{M}, \boldsymbol{\theta}_v) = \left(S_v^{(1)}(\tilde{\mathbf{u}}; \mathbf{M}, \boldsymbol{\theta}_v), \dots, S_v^{(L)}(\tilde{\mathbf{u}}; \mathbf{M}, \boldsymbol{\theta}_v)\right), \end{aligned}$$

with  $\boldsymbol{\theta}_u = (p_1, \dots, p_K, \lambda_1)$  and  $\boldsymbol{\theta}_v = (q_1, \dots, q_L, \lambda_2)$ . A proof of these equations is given in [16].

3) *Sparse Group PLS*: One potential drawback of gPLS is that it includes a group in the model only when all individual weights in that group are non-zero. However, sometimes we would like to combine both sparsity of groups and within each group. For example, if the predictor matrix contains genes, we might be interested in identifying particularly important genes in pathways of interest. The sparse group lasso [51] achieves this within group sparsity. The sparse group selection in the PLS methodology is accomplished using the sparse group lasso penalty [51] in the optimisation problem (6) and (7):

$$\begin{aligned} P_{\lambda_1}(\tilde{\mathbf{u}}) &= (1 - \alpha_1) \lambda_1 \sum_{k=1}^K \sqrt{p_k} \|\tilde{\mathbf{u}}^{(k)}\|_2 + \alpha_1 \lambda_1 \|\tilde{\mathbf{u}}\|_1, \\ P_{\lambda_2}(\tilde{\mathbf{v}}) &= (1 - \alpha_2) \lambda_2 \sum_{l=1}^L \sqrt{q_l} \|\tilde{\mathbf{v}}^{(l)}\|_2 + \alpha_2 \lambda_2 \|\tilde{\mathbf{v}}\|_1. \end{aligned}$$

The sparse group penalty introduces tuning parameters  $\alpha_2$  and  $\alpha_1$  which provide a link between the group lasso penalty ( $\alpha_1 = 0$ ,  $\alpha_2 = 0$ ) and the lasso ( $\alpha_1 = 1$ ,  $\alpha_2 = 1$ ). Depending on the combination of  $\alpha_1$  and  $\lambda_1$  (or  $\alpha_2$  and  $\lambda_2$ ) the (unscaled) weight subvector  $\tilde{\mathbf{u}}^{(k)}$  (or  $\tilde{\mathbf{v}}^{(l)}$ ) will be eliminated entirely, or sparsely estimated. The adaptation of the sparse group penalty for the PLS method was first considered in [16]. The closed form solution of the sparse group PLS method for the  $k$ -th subvector of  $\tilde{\mathbf{u}}^{(k)}$  is given by

$$S_u^{(k)}(\tilde{\mathbf{v}}; \mathbf{M}, \boldsymbol{\theta}_u) = \begin{cases} \mathbf{0} & \text{if } \frac{\|g_1\|_2}{(1-\alpha_1)\sqrt{p_k}} \leq \lambda_1 \\ \frac{g_1}{2} - \frac{\lambda_1(1-\alpha_1)\sqrt{p_k}g_1}{2\|g_1\|} & \text{otherwise} \end{cases}$$

where  $g_1 = g^{\text{soft}}(\mathbf{M}^{(k,\cdot)} \tilde{\mathbf{v}}, \lambda_1 \alpha_1 / 2)$ . Similarly, the  $l$ -th subvector of  $\tilde{\mathbf{v}}$  is given by

$$S_v^{(l)}(\tilde{\mathbf{u}}; \mathbf{M}, \boldsymbol{\theta}_v) = \begin{cases} \mathbf{0} & \text{if } \frac{\|g_2\|_2}{(1-\alpha_2)\sqrt{q_l}} \leq \lambda_2 \\ \frac{g_2}{2} - \frac{\lambda_2(1-\alpha_2)\sqrt{q_l}g_2}{2\|g_2\|} & \text{otherwise} \end{cases}$$

where  $g_2 = g^{\text{soft}}(\mathbf{M}^{(\cdot,l)} \tilde{\mathbf{u}}, \lambda_2 \alpha_2 / 2)$ . The sparsifyer functions for these penalties are:

$$\begin{aligned} \tilde{\mathbf{u}} &= S_u(\tilde{\mathbf{v}}; \mathbf{M}, \boldsymbol{\theta}_u) = \left(S_u^{(1)}(\tilde{\mathbf{v}}; \mathbf{M}, \boldsymbol{\theta}_u), \dots, S_u^{(K)}(\tilde{\mathbf{v}}; \mathbf{M}, \boldsymbol{\theta}_u)\right) \\ \tilde{\mathbf{v}} &= S_v(\tilde{\mathbf{u}}; \mathbf{M}, \boldsymbol{\theta}_v) = \left(S_v^{(1)}(\tilde{\mathbf{u}}; \mathbf{M}, \boldsymbol{\theta}_v), \dots, S_v^{(L)}(\tilde{\mathbf{u}}; \mathbf{M}, \boldsymbol{\theta}_v)\right) \end{aligned}$$

with  $\boldsymbol{\theta}_u = (p_1, \dots, p_K, \lambda_1, \alpha_1)$  and  $\boldsymbol{\theta}_v = (q_1, \dots, q_L, \lambda_2, \alpha_2)$ .

4) *Other penalties*: The penalties discussed so far have enforced general sparsity or sparsity with respect to a known grouping structure in the data. Extensions to the group structured sparsity in partial least squares setting have also been considered; in terms of overlapping groups [52], or additional grouping restrictions [53]. The penalisations considered so far all have all resulted in closed form solutions for the updates of  $\mathbf{u}$  and  $\mathbf{v}$ . We note here that this is not always the case. The fused Lasso penalty [54] is defined by:

$$\begin{aligned} P_{\lambda_1}(\tilde{\mathbf{u}}) &= \lambda_1 \alpha_1 \sum_{i=2}^p |\tilde{u}_i - \tilde{u}_{i-1}| + (1 - \alpha_1) \lambda_1 \|\tilde{\mathbf{u}}\|_1, \\ P_{\lambda_2}(\tilde{\mathbf{v}}) &= \lambda_1 \alpha_1 \sum_{i=2}^q |\tilde{v}_i - \tilde{v}_{i-1}| + (1 - \alpha_1) \lambda_1 \|\tilde{\mathbf{v}}\|_1. \end{aligned}$$

The first term in this penalty causes neighbouring coefficients to shrink together and will cause some to be identical, and the second causes regular Lasso shrinkage of the parameters for variable selection. Unlike the previous methods, a closed form solution for the fused Lasso cannot be directly achieved. This is because the penalty is not a separable function of the coordinates. Because there is no closed form solution for the fused Lasso, we cannot write a sparsifyer function so we have not considered this method. We note that methods exist that are able to solve the fused Lasso problem, either by reparameterisation, dynamic programming or path based algorithms. In particular, [23] have considered solving problems of the form (6) and (7) with the fused Lasso penalty. In their paper, they propose a sparse and fused penalised CCA, however in their derivation they assume  $\mathbf{X}^T \mathbf{X} = \mathbf{I}$  and  $\mathbf{Y}^T \mathbf{Y} = \mathbf{I}$ . In our framework, this method would be sparse and fused penalised PLS-SVD.

#### IV. THE UNIFIED ALGORITHM

Algorithm 1 permits to compute in a unified way, all four PLS versions (i)–(iv), with a possibility to add sparsity. Adjusted weights can also be computed and, if the number of requested components  $H$  is greater than 1, a deflation step is executed. Note that, if  $\mathbf{Y}$  is taken equal to  $\mathbf{X}$ , this algorithm performs Principal Component Analysis (PCA), as well as sparse PCA versions. If this is the case, the optimized criteria are simply restated in terms of variance instead of covariance.

##### ALGORITHM 1 HERE

**Remark 5.** On line 10, we impose that  $\|\mathbf{u}_1\|_2 = \|\mathbf{v}_1\|_2 = 1$  and  $u_{1,i} > 0$  where  $i = \operatorname{argmax}_{1 \leq j \leq p} |u_{1,j}|$  to ensure uniqueness of the results.

Note that  $\mathbf{w}_h$  and  $\mathbf{z}_h$  of lines 27, 29 and 32 correspond to  $\tilde{\mathbf{w}}_h$  and  $\tilde{\mathbf{z}}_h$  in the text.

**Algorithm 1** Sparse PLS algorithm for the four cases (i)–(iv)**Require:**  $\lambda_x, \lambda_y, H, X_0 = X, Y_0 = Y, \theta_u, \theta_v$ 

```

1:  $M_0 \leftarrow X_0^T Y_0$  ▷ Initialisation
2:  $P \leftarrow I$  and  $Q \leftarrow I$ 
3:  $u_0 \leftarrow c_0 \leftarrow \mathbf{0}_p, v_0 \leftarrow \mathbf{0}_q$  and  $\xi_0 \leftarrow \omega_0 \leftarrow \mathbf{1}_n$ 
4: If Case (iii) then
5:    $A \leftarrow (X_0^T X_0 + \lambda_x I)^{-1/2}$ 
6:    $B \leftarrow (Y_0^T Y_0 + \lambda_y I)^{-1/2}$ 
7:    $M_0 \leftarrow A M_0 B$ 
8: end if
9: for  $h = 1, \dots, H$  do
10:  Apply the SVD to  $M_{h-1}$  and extract the first
11:  triplet  $(\delta_1, u_1, v_1)$  of singular value and vectors.
12:  Set  $u_h \leftarrow u_1$  and  $v_h \leftarrow v_1$ 
13:  while convergence(*) of  $u_h$  do ▷ Sparsity step
14:     $\tilde{u}_h \leftarrow S_u(v_h; M_{h-1}, \theta_u)$ 
15:     $u_h \leftarrow \tilde{u}_h / \|\tilde{u}_h\|_2$ 
16:     $\tilde{v}_h \leftarrow S_v(u_h; M_{h-1}, \theta_v)$ 
17:     $v_h \leftarrow \tilde{v}_h / \|\tilde{v}_h\|_2$ 
18:  end while ▷ End of sparsity step
19:   $\xi_h \leftarrow X_{h-1}^T u_h$  ▷ X-score
20:   $\omega_h \leftarrow Y_{h-1}^T v_h$  ▷ Y-score
21:  If Case (i) then ▷ Adjusted weights step
22:     $w_h \leftarrow u_h$  and  $z_h \leftarrow v_h$ 
23:  end if
24:  If Case (ii) then
25:     $P \leftarrow P(I - u_{h-1} \xi_{h-1}^T X / \|\xi_{h-1}\|^2)$ 
26:     $Q \leftarrow Q(I - v_{h-1} \omega_{h-1}^T Y / \|\omega_{h-1}\|^2)$ 
27:     $w_h \leftarrow P u_h$  and  $z_h \leftarrow Q v_h$ 
28:  end if
29:  If Case (iii) then  $w_h \leftarrow A u_h$  and  $z_h \leftarrow B v_h$ 
30:  If Case (iv) then
31:     $P \leftarrow P(I - u_{h-1} c_{h-1}^T)$ 
32:     $w_h \leftarrow P u_h$ 
33:     $z_h \leftarrow v_h$ 
34:  end if ▷ End of adjusted weights step
35:  If Case (i) or (iii) then ▷ Deflation step
36:     $c_h^T \leftarrow u_h^T$  and  $e_h^T \leftarrow v_h^T$ 
37:  end if
38:  If Case (ii) or (iv) then  $c_h^T \leftarrow \xi_h^T X_{h-1} / \|\xi_h\|_2^2$ 
39:  If Case (ii) then  $e_h^T \leftarrow \omega_h^T Y_{h-1} / \|\omega_h\|_2^2$ 
40:  If Case (iv) then  $d_h^T \leftarrow \xi_h^T Y_{h-1} / \|\xi_h\|_2^2$ 
41:   $X_h \leftarrow X_{h-1} - \xi_h c_h^T$ 
42:  If Case (iv) then
43:     $Y_h \leftarrow Y_{h-1} - \xi_h d_h^T$  ▷ PLS2
44:  Else
45:     $Y_h \leftarrow Y_{h-1} - \omega_h e_h^T$ 
46:  End If
47:   $M_h \leftarrow X_h^T Y_h$  ▷ End of deflation step
48: Store  $\xi_h, \omega_h, u_h, v_h, w_h, z_h$ 
49: end for

```

(\*) Convergence of a vector  $t$  is tested on the change in  $t$ , i.e.,  $\|t_{\text{old}} - t_{\text{new}}\| / \|t_{\text{new}}\| < \epsilon$ , where  $\epsilon$  is “small”, e.g.,  $10^{-8}$ .

At this point, it is worthwhile noting that that when  $p$  and  $q$  are small compared to  $n$ , one can slightly modify Algorithm 1 by using the recursive equations that express  $M_h$  in terms of  $M_{h-1}$ , instead of using the recursions on  $X_h$  and  $Y_h$ . The former are provided in subsection II-E. This should increase speed of execution of the algorithm.

Moreover, one can use various approaches to deal with the cases when  $n, p$  or  $q$  are too large in our algorithm, making some objects not fittable into the computer’s memory. These can be divided into *chunk approaches* and *streaming (or incremental) approaches*, which are presented in the next subsections. Of course, any combinations of these approaches can be used if necessary. Some of these approaches might even increase the computation speed, even in a context where all objects would fit into memory.

**A. Matrix multiplication using chunks**

To scale Algorithm 1 to big data (i.e., very large  $n \gg p$  and  $q$ ), we can use a simple idea to multiply two very large matrices that are too big to fit into the computer’s memory.

Let us divide the total number  $n$  of rows of  $X$  (resp. of  $Y$ ) into blocks  $X_{(g)}$  (resp.  $Y_{(g)}$ ),  $g = 1, \dots, G$ , of (approximatively) the same size. We have

$$X^T Y = \sum_{g=1}^G X_{(g)}^T Y_{(g)}.$$

The number of blocks  $G$  has to be chosen so that each product  $X_{(g)}^T Y_{(g)} : p \times q$  can be done within the available RAM. Note that all these products can be performed in parallel if the required computing equipment is available.

**B. SVD when  $p$  or  $q$  is very large**

The main step of our algorithm is the computation of the first triplet  $(\delta_1, u_1, v_1)$  in the SVD of the  $(p \times q)$  matrices  $M_{h-1}$ . The `irlba` [55] R package can be used to compute quite easily this triplet for values of  $p$  and  $q$  as big as 50,000. This package is based on an augmented implicitly restarted Lanczos bidiagonalization method [56].

When  $p$  (or  $q$ ) is much larger, another approach is necessary to compute the SVD of  $M_{h-1}$ ; see e.g., [57]. Suppose that  $p$  is large but not  $q$ , which is common in several applications. We thus suppose that  $p \gg q$ . The Algorithm 1 in [57] is now presented to highlight the elements needed in our algorithm. We can partition a large matrix  $M : p \times q$  by rows into a small number  $s$  of submatrices (or chunks):

$$M = \begin{pmatrix} M_1 \\ M_2 \\ \vdots \\ M_s \end{pmatrix}.$$

Let  $M_i = U_i D_i V_i^T$  denote the SVD of  $M_i : g \times q$  such that  $gs = p$  (w.l.o.g.). We can take  $g$  much larger than  $q$  as long as

it is still possible to compute the SVD of these submatrices. Define

$$\tilde{U} = \begin{pmatrix} U_1 & & & \\ & U_2 & & \\ & & \ddots & \\ & & & U_s \end{pmatrix} : p \times p$$

where  $U_i : g \times g$  and define

$$H = \begin{pmatrix} D_1 V_1^\top \\ D_2 V_2^\top \\ \vdots \\ D_s V_s^\top \end{pmatrix} : p \times q,$$

where  $D_i : g \times q$  and  $V_i : q \times q$ , so that  $M = \tilde{U}H$ . Let  $H = U_H D_H V_H^\top$  be the SVD of  $H$ . Note that this matrix is as large as  $M$  so one may wonder what has been gained with this approach. But  $D_i$  being a  $g \times q$  diagonal rectangular matrix,  $D_i V_i^\top$  has  $g - q$  zero row-vectors in its bottom. Consequently, the matrix  $H$  contains only  $sq$  non-zero row vectors. Now let  $\tilde{H} = RH : p \times q$  be a rearrangement in rows for  $H$  such that its first  $sq$  row vectors are non-zero and  $p - sq$  row vectors are in its bottom. We now have to compute  $U^* D^* V^{*\top}$ , the SVD of a (much smaller)  $sq \times q$  matrix<sup>1</sup>:

$$\begin{aligned} H &= R^\top \tilde{H} \\ &= R^\top \begin{bmatrix} U^* D^* V^{*\top} \\ 0 \end{bmatrix} \\ &= R^\top \underbrace{\begin{bmatrix} U^* & 0 \\ 0 & I_{p-sq} \end{bmatrix}}_{U_H} \underbrace{\begin{bmatrix} D^* \\ 0 \end{bmatrix}}_{D_H} \underbrace{V^{*\top}}_{V_H^\top}. \end{aligned}$$

We obtain

$$M = (\tilde{U}U_H)D_H V_H^\top$$

which forms a SVD of  $M$ .

Now, let  $\mathbf{1}$  be a vector containing only 0s but a 1 in the first position. For our PLS algorithm, we only need to compute the first triplet in the SVD of  $M$ , namely  $\delta_1 = \mathbf{1}_p^\top D_H \mathbf{1}_q = D_{1,1}^*$ ,  $\mathbf{v}_1 = V_{\bullet,1} = V_H \mathbf{1}_q = V^* \mathbf{1}_q$  and the first column of  $(\tilde{U}U_H)$ :

$$\begin{aligned} \mathbf{u}_1 = (\tilde{U}U_H)\mathbf{1}_p &= \tilde{U}R^\top \begin{bmatrix} U^* & 0 \\ 0 & I_{p-sq} \end{bmatrix} \mathbf{1}_p \\ &= \tilde{U}R^\top \begin{bmatrix} U^* \mathbf{1}_{sq} \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} U_{1,\bullet q}(U_{\bullet,1}^*)_{1,\dots,q} \\ U_{2,\bullet q}(U_{\bullet,1}^*)_{q+1,\dots,2q} \\ \vdots \\ U_{s,\bullet q}(U_{\bullet,1}^*)_{(s-1)q+1,\dots,sq} \end{bmatrix}. \end{aligned}$$

It is seen above that only the  $q$  first triplets of the SVDs of the  $M_i$ s are required. So, overall we “only” have to compute  $s$  truncated ( $q \times q$ ) SVDs (of the  $M_i$ s) and one truncated ( $1 \times 1$ ) SVD (of the  $sq$  first lines of  $\tilde{H}$ , which are easily obtained from these truncated SVDs).

Moreover, we can even compute  $\mathbf{u}_1$  from  $\mathbf{v}_1$  using the simple formula  $\mathbf{u}_1 = M\mathbf{v}_1 / \|M\mathbf{v}_1\|$  (using a chunk

approach).

When  $q$  is larger than  $p$ , we just partition  $M$  in columns instead of rows. When both  $p$  and  $q$  are large, one can adapt Algorithm 2 in [57] which generalizes the above. (They even propose a third algorithm for the case of online (streaming) SVDs.)

Note that these algorithms based on the split-and-merge strategy possess an embarrassingly parallel structure and thus can be efficiently implemented on a distributed or multicore machine.

### C. Incremental SVD when $n$ is large

We want to compute the truncated SVD of the matrix  $M_h = X_h^\top Y_h$  when  $n$  is very large (and the  $X$ - and  $Y$ -matrices are split in blocks, or chunks, of size  $n/G$  for some given  $G$ ). One can use the divide and conquer approach presented in subsection A to compute first the matrix  $M_h = X_h^\top Y_h$  and then evaluate the SVD of this matrix. We present here an alternative approach [58] by considering an incremental version of the SVD.

Let  $X^\top = [x_1^\top, \dots, x_n^\top]$  and  $Y^\top = [y_1^\top, \dots, y_n^\top]$  be non-centered data matrices. We note

$$M_n = \dot{X}_n^\top \dot{Y}_n = \sum_{i=1}^n (x_i - \mu_{X,n})(y_i - \mu_{Y,n})^\top$$

with the centered data matrices

$$\dot{X}_n = X_n - \mathbf{1}_n \mu_{X,n}^\top, \quad \dot{Y}_n = Y_n - \mathbf{1}_n \mu_{Y,n}^\top$$

where

$$\mu_{X,n} = n^{-1} X^\top \mathbf{1}_n = n^{-1} \sum_{i=1}^n x_i^\top$$

and

$$\mu_{Y,n} = n^{-1} Y^\top \mathbf{1}_n = n^{-1} \sum_{i=1}^n y_i^\top.$$

We have the streaming updating formulas

$$\begin{aligned} \mu_{X,n+1} &= \frac{n}{n+1} \mu_{X,n} + \frac{1}{n+1} x_{n+1}^\top, \\ \mu_{Y,n+1} &= \frac{n}{n+1} \mu_{Y,n} + \frac{1}{n+1} y_{n+1}^\top, \end{aligned}$$

and

$$M_{n+1} = M_n + \frac{n}{(n+1)} (x_{n+1} - \mu_{X,n})(y_{n+1} - \mu_{Y,n})^\top. \quad (14)$$

Now, let the  $H$ -rank truncated SVD of  $M_n$  be  $M_n^{(H)} = U_{n,\bullet H} \Delta_{n,H} V_{n,\bullet H}^\top$ . Let  $\tilde{x}_{n+1} = x_{n+1} - \mu_{X,n}$  and  $\tilde{y}_{n+1} = y_{n+1} - \mu_{Y,n}$ . Since  $U_{n,\bullet H}^\top U_{n,\bullet H} = I$ , we have

$$\begin{aligned} \tilde{x}_{n+1} &= \mathcal{P}_{U_{n,\bullet H}} \tilde{x}_{n+1} + \mathcal{P}_{U_{n,\bullet H}^\perp} \tilde{x}_{n+1} \\ &= U_{n,\bullet H} U_{n,\bullet H}^\top \tilde{x}_{n+1} + \mathcal{P}_{U_{n,\bullet H}^\perp} \tilde{x}_{n+1} \\ &= U_{n,\bullet H} c_{n+1} + \tilde{x}_{n+1}^\perp \end{aligned}$$

with  $c_{n+1} = U_{n,\bullet H}^\top \tilde{x}_{n+1}$  and  $\tilde{x}_{n+1}^\perp = \mathcal{P}_{U_{n,\bullet H}^\perp} \tilde{x}_{n+1}$ . Similarly,

$$\tilde{y}_{n+1} = V_{n,\bullet H} d_{n+1} + \tilde{y}_{n+1}^\perp$$

<sup>1</sup>The transpose sign on  $R$  is missing in [57].

with  $\mathbf{d}_{n+1} = \mathbf{V}_{n,\bullet H}^\top \tilde{\mathbf{y}}_{n+1}$  and  $\tilde{\mathbf{y}}_{n+1}^\perp = \mathcal{P}_{\mathbf{V}_{n,\bullet H}^\perp} \tilde{\mathbf{y}}_{n+1}$ . Now, in view of (14), we have the approximation

$$\mathbf{M}_{n+1}^{(H)} \approx \mathbf{M}_n^{(H)} + \frac{n}{n+1} \tilde{\mathbf{x}}_{n+1} \tilde{\mathbf{y}}_{n+1}^\top.$$

**Remark 6.** Note that this approximation is in fact exact when  $H = \text{rank}(\mathbf{M}_n)$ . So if we want to use this approach in our algorithm, we would have to compute all the singular elements and not only the first triplet. This being said, if for example  $q$  is not too large (e.g.,  $q = 1$ ) this is not a problem anymore. Moreover, we see from Appendix A-A that  $\mathbf{u}_1 = \mathbf{X}^\top \mathbf{Y} \mathbf{v}_1 / \|\mathbf{X}^\top \mathbf{Y} \mathbf{v}_1\|$  and  $\mathbf{v}_1 = \mathbf{Y}^\top \mathbf{X} \mathbf{u}_1 / \|\mathbf{Y}^\top \mathbf{X} \mathbf{u}_1\|$ . Note also that  $\mathbf{u}_1$  is the first eigenvector of the  $p \times p$  matrix  $(\mathbf{Y}^\top \mathbf{X})^\top \mathbf{Y}^\top \mathbf{X}$  whereas  $\mathbf{v}_1$  is the first eigenvector of the  $q \times q$  matrix  $(\mathbf{X}^\top \mathbf{Y})^\top \mathbf{X}^\top \mathbf{Y}$ . So we only need to compute either  $\mathbf{u}_1$  (if  $p < q$ ) or  $\mathbf{v}_1$  (if  $q \leq p$ ), from which we obtain the other one.

At this point, one can write

$$\mathbf{M}_{n+1}^{(H)} = \left[ \mathbf{U}_{n,\bullet H}, \frac{\tilde{\mathbf{x}}_{n+1}^\perp}{\|\tilde{\mathbf{x}}_{n+1}^\perp\|_2} \right] \mathbf{Q}_{n+1} \left[ \mathbf{V}_{n,\bullet H}, \frac{\tilde{\mathbf{y}}_{n+1}^\perp}{\|\tilde{\mathbf{y}}_{n+1}^\perp\|_2} \right]^\top$$

with

$$\mathbf{Q}_{n+1} = \frac{n}{n+1} \begin{pmatrix} \frac{n+1}{n} \Delta_n + \mathbf{c}_{n+1} \mathbf{d}_{n+1}^\top & \frac{\|\tilde{\mathbf{y}}_{n+1}^\perp\|_2 \mathbf{c}_{n+1}}{\|\tilde{\mathbf{x}}_{n+1}^\perp\|_2 \|\tilde{\mathbf{y}}_{n+1}^\perp\|_2} \\ \frac{\|\tilde{\mathbf{x}}_{n+1}^\perp\|_2 \mathbf{d}_{n+1}^\top}{\|\tilde{\mathbf{x}}_{n+1}^\perp\|_2 \|\tilde{\mathbf{y}}_{n+1}^\perp\|_2} & \frac{\|\tilde{\mathbf{y}}_{n+1}^\perp\|_2 \mathbf{c}_{n+1}}{\|\tilde{\mathbf{x}}_{n+1}^\perp\|_2 \|\tilde{\mathbf{y}}_{n+1}^\perp\|_2} \end{pmatrix}.$$

It then suffices to perform the SVD of the matrix  $\mathbf{Q}_{n+1}$  of dimension  $(H+1) \times (H+1)$ . Writing  $\mathbf{Q}_{n+1} = \mathbf{A}_{n+1} \mathbf{S}_{n+1} \mathbf{B}_{n+1}^\top$ , we have

$$\mathbf{M}_{n+1}^{(H)} = \mathbf{U}_{n+1} \Delta_{n+1} \mathbf{V}_{n+1}^\top$$

with  $\Delta_{n+1} = \mathbf{S}_{n+1}$ ,

$$\mathbf{U}_{n+1} = \left[ \mathbf{U}_n, \frac{\tilde{\mathbf{x}}_{n+1}^\perp}{\|\tilde{\mathbf{x}}_{n+1}^\perp\|_2} \right] \mathbf{A}_{n+1}$$

and

$$\mathbf{V}_{n+1} = \left[ \mathbf{V}_n, \frac{\tilde{\mathbf{y}}_{n+1}^\perp}{\|\tilde{\mathbf{y}}_{n+1}^\perp\|_2} \right] \mathbf{B}_{n+1}.$$

To keep the approximation  $\mathbf{M}_{n+1}^{(H)}$  of  $\mathbf{M}_{n+1}$  at rank  $H$ , the row and column of  $\Delta_{n+1}$  containing the smallest singular value are deleted and the associated singular vectors are deleted from  $\mathbf{U}_{n+1}$  and  $\mathbf{V}_{n+1}$ .

This incremental way to compute the SVD provides a promising alternative for handling very large sample size (specially when  $q$  is not too large). Moreover the incremental SVD is well designed in a data stream context.

## V. NUMERICAL EXPERIMENTS

In this section we use the R software to carry out a short simulation study in order to illustrate the numerical behaviour of the new proposed approach. The experiments have been conducted using a laptop with a 2.53 GHz processor and 8 GB of memory. The parallel strategy utilizes four processor cores.

We present two simulations to illustrate the good performance of the proposed approaches and the scalability to large

sample sizes of our algorithm. The first simulation considers the PLS-R model (case (iv)) on group structure data while the second simulation presents an extension of PLS approaches to discriminant analysis purpose.

### A. Group PLS model

We generate data with a group structure: 20 groups of 20 variables for  $\mathbf{X}$  ( $p = 400$ ) and 25 groups of 20 variables for  $\mathbf{Y}$  ( $q = 500$ ). To highlight the scalability of our algorithm, we generate two big matrices from the following models linked by  $H = 2$  latent variables:

$$\mathbf{X} = \Xi_H \mathbf{C}_H^\top + \mathbf{F}_H^X, \quad \mathbf{Y} = \Xi_H \mathbf{D}_H^\top + \mathbf{F}_H^Y, \quad (15)$$

where the matrix  $\Xi_H = (\xi_j)$  contains 2 latent variables  $\xi_1$  and  $\xi_2$ . The entries in these vectors have all been independently generated from a standard normal distribution. The rows of the residual matrix  $\mathbf{F}_H^X$  (respectively,  $\mathbf{F}_H^Y$ ) have been generated from a multivariate normal distribution with zero mean  $\mu_X$  (resp.  $\mu_Y$ ) and covariance matrix  $\Sigma_X = 1.5^2 \mathbf{I}_p$  (resp.  $\Sigma_Y = 1.5^2 \mathbf{I}_q$ ).

Among the 20 groups of  $\mathbf{X}$ , only 4 groups each containing 15 true variables and 5 noise variables are associated to the response variables of  $\mathbf{Y}$ . We set the  $p$ -vector  $\mathbf{c}_1$  (first column of the  $\mathbf{C}_H$  matrix) to have 15 1s, 30 -1s and 15 1.5s, the other entries being all set to 0. All 15 non-zero coefficients are assigned randomly into one group along with the remaining 5 zero coefficients corresponding to noise variables. The vector  $\mathbf{c}_2$  is chosen in the same way as  $\mathbf{c}_1$ . The two columns of  $\mathbf{D}_H$  are  $q$ -vectors containing 15 -1s, 15 -1.5s and 30 1s and the rest are 0s such that the matrix  $\mathbf{Y}$  have a similar group structure for 4 groups containing the signal. Finally, the sample size is set to  $n = 560,000$  observations which corresponds to storage requirements of approximately 5 GB for each matrix, thus with a total exceeding the 8 GB of memory available on our laptop.

The top four plots of Figure 1 show the results of the group PLS estimated with only  $n = 100$  observations. For such a sample size, the usual group PLS can be used without any computational time or memory issues. In this case, group PLS manages to select the relevant groups and performs well to estimate the weight vectors  $\mathbf{u}_1$  and  $\mathbf{v}_1$  related to the first component and the weight vectors  $\mathbf{u}_2$  and  $\mathbf{v}_2$  related to the second component.

The bottom four plots of Figure 1 show the results of the group PLS estimated on the full data set which can be only analyzed by using the extended version of our algorithm for big data. In this run, we use  $G = 100$  chunks for enabling matrix multiplication. The execution time was around 15 minutes for two components ( $H = 2$ ) and took less than 2 minutes for getting the first component. We can observe that the signal has been perfectly identified and estimated, which is expected for such a huge amount of information.

Note that for validation purposes, the extended version of our algorithm for big data have been ran and gave exactly the same results than the usual algorithm on the small data set ( $n = 100$ ).

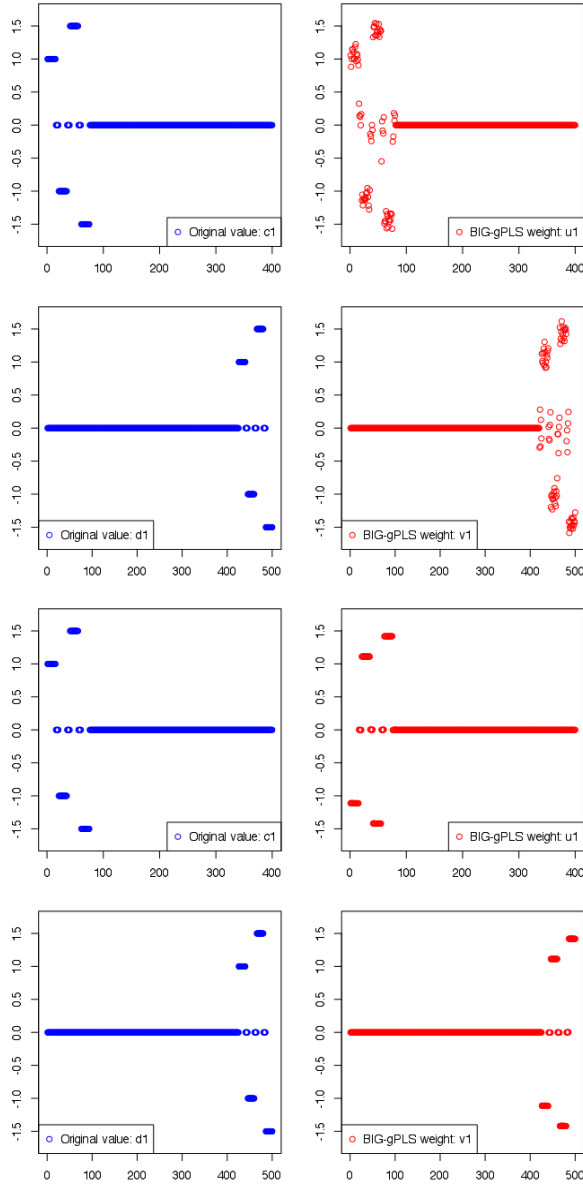


Fig. 1. Comparison of the signal recovered (weights  $u_1$  and  $v_1$ ) by the first component ( $H = 2$ ) of the gPLS. For the top four plots,  $n = 100$ , and for the four bottom plots  $n = 560,000$ . Left column: the true values of  $c_1$  and  $d_1$  for small and large sample sizes. Right column: the estimated values of  $u_1$  and  $v_1$  for small and large sample sizes. Note that the values of  $u_1$  (resp.  $v_1$ ) have been rescaled so that its norm equals that of the original  $c_1$  (resp.  $d_1$ )

### B. Case of Regularised PLS-DA

We consider here the case of qualitative response variables for discrimination analysis. In this framework, PLS approaches have often been used [59] by recoding the qualitative response as a dummy block matrix  $Y : n \times c$  indicating the class of each sample ( $c$  being the number of categories). One can also directly apply PLS regression on the data as if  $Y$  was a matrix with continuous entries (from now on called PLS-DA). Note that [60] give some theoretical justification for this approach. A group and a sparse group version have been proposed by [16] using only penalties on the loading related to the variables

in  $X$ . Our unified algorithm is then naturally extended in the same way to deal with categorical variables. We illustrate it on a big data set defined as follows. Let  $A_k$  be the set of indices  $(i, j)$  of the  $i$ -th observation and  $j$ -th variable that are associated to the corresponding grey cell as shown in Figure 2.  $\forall k = 1, \dots, 6, \forall i = 1, \dots, n, \forall j = 1, \dots, p$

$$X_{i,j} = \mu_k \times 1_{\{(i,j) \in A_k\}} + \epsilon_{i,j}$$

where  $\mu^T = (\mu_1, \dots, \mu_6) = (-1.0, 1.5, 1.0, 2.5, -0.5, 2.0)$ , and  $\epsilon_{i,j} \sim N(0, 1)$ . As illustrated on Figure 2, the matrix  $X$  is composed of 6 groups of  $p_k = 100$  variables ( $p = \sum_{k=1}^6 p_k = 600$ ) and each of the 3 categories of the response variable are linked to two groups of variables. We used a sample size of  $n = 486,000$  which corresponds to storage requirements of approximately 5 GB for the  $X$  matrix. We use  $G = 100$  chunks for computing the different matrix products. The run took around 9 minutes for a model using 2 components. The relevant groups have been selected in both components. We randomly sample 9,000 observations and present in Figure 3 their projection on the two components estimated on the full data set. A nice discrimination of the 3 categories of the response variable is observed.

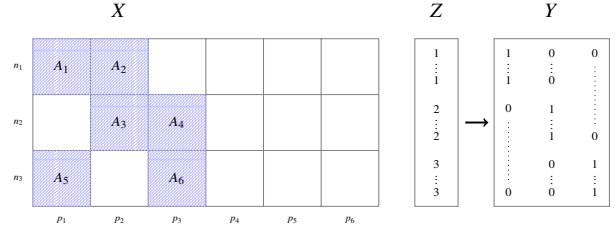


Fig. 2. Discriminant Analysis Design Matrices

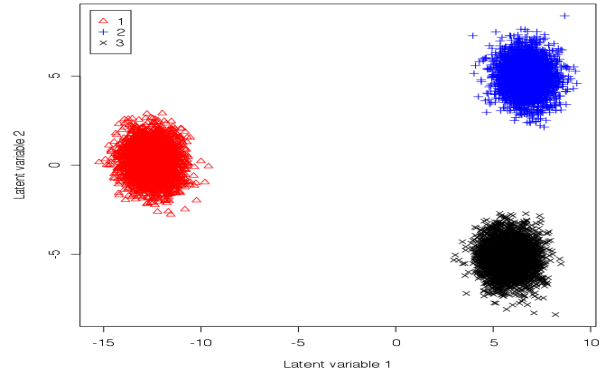


Fig. 3. Group PLS-DA on a big data set

## VI. CONCLUSION AND FUTURE WORK

This paper surveys four popular partial least squares methods, and unifies these methods with recent variable selection techniques based on penalised singular value decomposition. We present a general framework for both symmetric and asymmetric penalised PLS methods and showcase some possible convex penalties. A unified algorithm is described and implemented for the penalised PLS methods, and we offer further extensions to deal with massive data sets ( $n$ ,  $p$  and  $q$

very large). A full comparison in terms of time and memory of the different proposed extensions is an open area of future research.

Aside from computational issues, it is unclear if retaining the deflations of the usual PLS methods is appropriate when there is penalisation. In particular, we note that the orthogonality constraints of the original PLS methods are not retained for the penalised methods. Further development of our methods could seek to preserve the orthogonality constraints. We are perusing this open area using ideas from [61], and [47] for the simple lasso penalty. However, further investigation is required in the context of more complex penalties such as group or sparse group penalties.

## APPENDIX A

### PROOFS OF SOME RESULTS

#### A. Proof of (C1) in subsection II-D

The proof is given here for completeness. It follows the lines of [62, example Sec. 2.4].

Imposing  $\|u\|_2 = \|v\|_2 = 1$  we have

$$\|M - \delta uv^T\|_F^2 = \|M\|_F^2 - 2\delta u^T M v + \delta^2.$$

Since  $M$  is fixed, the minimisation problem is equivalent to

$$\underset{\|u\|_2=\|v\|_2=1, \delta>0}{\text{minimise}} \quad -2\delta u^T M v + \delta^2$$

subject to  $u^T u_j = v^T v_j = 0, 1 \leq j < h$ . The Lagrangian is

$$L = -2\delta u^T M v + \delta^2 - \alpha(u^T u - 1) - \beta(v^T v - 1) - \sum_{j=1}^{h-1} \mu_j u^T u_j - \sum_{j=1}^{h-1} \nu_j v^T v_j$$

with Lagrangian multipliers  $\alpha, \beta, \mu_j, \nu_j$  for  $1 \leq j < h$ . Now

$$\frac{\partial}{\partial \delta} L = -2u^T M v + 2\delta$$

which should be equal to 0 at the optimum, leading to

$$\delta = u^T M v.$$

Substituting this  $\delta$  into the optimisation function gives

$$\underset{\|u\|_2=\|v\|_2=1}{\text{minimise}} \quad -(u^T M v)^2$$

subject to  $u^T u_j = v^T v_j = 0, 1 \leq j < h$ . Noting that  $u^T M v = \text{Cov}(Xu, Yv)$  and since we impose  $\delta > 0$ , this can be rewritten as

$$\underset{\|u\|_2=\|v\|_2=1}{\text{maximise}} \quad \text{Cov}(Xu, Yv)$$

subject to  $u^T u_j = v^T v_j = 0, 1 \leq j < h$ . We now consider the claim for the first pair of singular vectors

$$\underset{\|u\|_2=\|v\|_2=1}{\text{maximise}} \quad \text{Cov}(Xu, Yv),$$

with Lagrangian

$$L = u^T X^T Y v + \alpha(u^T u - 1) + \beta(v^T v - 1).$$

We have to solve

$$\begin{cases} \frac{\partial}{\partial u} L = X^T Y v + 2\alpha u = 0 \\ \frac{\partial}{\partial v} L = Y^T X u + 2\beta v = 0 \\ \frac{\partial}{\partial \alpha} L = u^T u - 1 = 0 \\ \frac{\partial}{\partial \beta} L = v^T v - 1 = 0 \end{cases}$$

(Note that  $v$  is proportional to  $Y^T X u$ .) We multiply the first equation by  $u^T$  and the second by  $v^T$ . This gives, using the third and the fourth,

$$\alpha = \beta = -2^{-1} u^T X^T Y v = -2^{-1} \text{Cov}(Xu, Yv).$$

We multiply the first equation by  $Y^T X$  and the second by  $X^T Y$ . This gives

$$Y^T X X^T Y v + 2\alpha Y^T X u = 0$$

and thus

$$(X^T Y)^T X^T Y v = 4\alpha \beta v.$$

Similarly

$$(Y^T X)^T Y^T X u = 4\alpha \beta u.$$

So  $u_1$  and  $v_1$  are (normed) eigenvectors respectively of  $(Y^T X)^T Y^T X$  and  $(X^T Y)^T X^T Y$  associated to the same eigenvalue ( $\lambda = 4\alpha\beta$ ). Now,

$$\text{Cov}^2(Xu, Yv) = \lambda,$$

so  $u_1$  and  $v_1$  must be the eigenvectors associated to the largest eigenvalue, noted  $\lambda_1$ . Also, we have to choose the sign of  $u_1$  (or  $v_1$ ) so that the covariance is maximal and positive. Now, for the remaining  $u_h$  and  $v_h$  ( $h > 1$ ), since they must also maximize the covariance (under some successive added orthogonality constraints), they also need to be eigenvectors associated to the same matrices  $(Y^T X)^T Y^T X$  and  $(X^T Y)^T X^T Y$ . It is clear that they are the eigenvectors associated to the remaining eigenvalues  $\lambda_2 > \dots > \lambda_{\min(p,q)}$ , and that

$$\sqrt{\lambda_h} = \text{Cov}(Xu_h, Yv_h)$$

if the sign of  $u_h$  (or  $v_h$ ) is set correctly. It is easy to conclude using the link between the SVD and the eigen decomposition that  $u_h$  and  $v_h$  are the singular vectors of  $X^T Y$ .

#### B. Proof of (C2) in subsection II-D

We want to find the successive pairs of vectors  $(\tilde{w}_1, \tilde{z}_1), \dots, (\tilde{w}_r, \tilde{z}_r)$  solution of

$$\underset{\tilde{w}, \tilde{z}}{\text{argmax}} \quad \text{Cor}(X\tilde{w}, Y\tilde{z}),$$

subject to the constraints  $\text{Cov}(X\tilde{w}, X\tilde{z}_j) = \text{Cov}(Y\tilde{w}, Y\tilde{z}_j) = 0, 1 \leq j < h$ .

Let  $u = (X^T X)^{1/2} \tilde{w}$  and  $v = (Y^T Y)^{1/2} \tilde{z}$ . We have

$$\begin{aligned} \text{Cor}(X\tilde{w}, Y\tilde{z}) &= \frac{\tilde{w}^T X^T Y \tilde{z}}{\sqrt{(\tilde{w}^T X^T X \tilde{w})(\tilde{z}^T Y^T Y \tilde{z})}} \\ &= \frac{u^T (X^T X)^{-1/2} X^T Y (Y^T Y)^{-1/2} v}{\sqrt{(u^T u)(v^T v)}}. \end{aligned}$$

Since the above expression is invariant to the scaling of  $u$  and  $v$ , the objective function is equivalent to maximising the covariance between the scores under the constraint that their variances is equal to 1. This is also equivalent to maximising

$$\underset{\|u\|_2=\|v\|_2=1}{\text{argmax}} \quad \text{Cov}(X(X^T X)^{-1/2} u, Y(Y^T Y)^{-1/2} v),$$

subject to the constraints

$$\begin{aligned} \text{Cov}(\mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{u}, \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{u}_j) &= \\ \text{Cov}(\mathbf{Y}(\mathbf{Y}^\top \mathbf{Y})^{-1/2} \mathbf{v}, \mathbf{Y}(\mathbf{Y}^\top \mathbf{Y})^{-1/2} \mathbf{v}_j) &= 0, \end{aligned}$$

$1 \leq j < h$ . But note that

$$\begin{aligned} \text{Cov}(\mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{u}_h, \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{u}_j) \\ = \mathbf{u}_h^\top (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{X}^\top \mathbf{X} (\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{u}_j = \mathbf{u}_h^\top \mathbf{u}_j. \end{aligned}$$

and similarly for  $\mathbf{v}$ . So we in fact want to solve

$$\underset{\|\mathbf{u}\|_2=\|\mathbf{v}\|_2=1}{\text{argmax}} \quad \text{Cov}(\mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{u}, \mathbf{Y}(\mathbf{Y}^\top \mathbf{Y})^{-1/2} \mathbf{v}),$$

subject to the constraints  $\mathbf{u}^\top \mathbf{u}_j = \mathbf{v}^\top \mathbf{v}_j = 0$ ,  $1 \leq j < h$ . Applying (C1), it is direct that they are the singular vectors of  $(\mathbf{X}^\top \mathbf{X})^{-1/2} \mathbf{X}^\top \mathbf{Y} (\mathbf{Y}^\top \mathbf{Y})^{-1/2}$ .

### C. Link between eigen elements and singular elements

Let

$$\mathbf{X} = \mathbf{U} \mathbf{D} \mathbf{V}^\top$$

be the singular decomposition of some matrix  $\mathbf{X}$ . Now,

$$\begin{aligned} \mathbf{X}^\top \mathbf{X} &= \mathbf{V} \mathbf{D} \mathbf{U}^\top \mathbf{U} \mathbf{D} \mathbf{V}^\top \\ &= \mathbf{V} \mathbf{D}^2 \mathbf{V}^\top. \end{aligned}$$

We recognize the eigenvalue decomposition of the matrix  $\mathbf{X}$ . Thus, it is clear that the eigenvalues of  $\mathbf{X}^\top \mathbf{X}$  are the squares of the singular values of  $\mathbf{X}$ , and that the eigenvectors of  $\mathbf{X}^\top \mathbf{X}$  are the right singular vectors of  $\mathbf{X}$ . Similarly for the left eigenlements:

$$\begin{aligned} \mathbf{X} \mathbf{X}^\top &= \mathbf{U} \mathbf{D} \mathbf{V}^\top \mathbf{V} \mathbf{D} \mathbf{U}^\top \\ &= \mathbf{U} \mathbf{D}^2 \mathbf{U}^\top. \end{aligned}$$

### D. The two versions of NIPALS: scaled/unscaled

From the (compact) SVD decomposition  $\mathbf{M}_{h-1} = \mathbf{X}_{h-1}^\top \mathbf{Y}_{h-1} = \mathbf{U}_h \mathbf{\Delta}_h \mathbf{V}_h^\top$ , we obtain  $\mathbf{V}_h = \mathbf{Y}_{h-1}^\top \mathbf{X}_{h-1} \mathbf{U}_h \mathbf{\Delta}_h^{-1}$  and thus  $\mathbf{v}_h = \delta_h^{-1} \mathbf{Y}_{h-1}^\top \mathbf{\xi}_h = (\|\mathbf{\xi}_h\|^{-2} \delta_h)^{-1} \mathbf{Y}_{h-1}^\top \mathbf{\xi}_h / \|\mathbf{\xi}_h\|^2$ , where  $\mathbf{v}_h$  is the first column of  $\mathbf{V}_h$  and  $\delta_h = \|\mathbf{Y}_{h-1}^\top \mathbf{\xi}_h\|$  is the first diagonal element of  $\mathbf{\Delta}_h$ . This vector  $\mathbf{v}_h$  is normed. This is exactly what is done in [41, p. 212, step 6] (despite an erroneous transpose sign). But this differs to the classic PLS2 algorithm [36] which follows the same process but does not include this scaling; see [11, p. 117] or [42, p. 128]. They instead compute, at each step  $h$ , a (not scaled) vector  $\mathbf{Y}_{h-1}^\top \mathbf{\xi}_h / (\mathbf{\xi}_h^\top \mathbf{\xi}_h)$ , which they note  $\mathbf{c}$  (not to be confounded with our  $\mathbf{c}_h$ ). It is proportional to our  $\mathbf{v}_h$ , with  $\mathbf{v}_h = (\mathbf{c}^\top \mathbf{c})^{-1/2} \mathbf{c} = \alpha_h \mathbf{c}$ , where  $\alpha_h := \|\mathbf{\xi}_h\|^2 / \|\mathbf{Y}_{h-1}^\top \mathbf{\xi}_h\|$ .

Now, define  $p_h = \alpha_h^{-1}$  in the scaled case and  $p_h = 1$  otherwise. The  $Y$ -score vectors are defined as  $\omega_h = p_h \alpha_h \mathbf{Y}_{h-1} \mathbf{v}_h$  (which is noted  $\mathbf{u}$  by the authors of the unscaled case).

For both algorithms, the fitted values  $\hat{\mathbf{Y}}_{(h)} = b_h \mathbf{\xi}_h \mathbf{v}_h^\top$  (or  $\hat{\mathbf{Y}}_{(h)} = b_h \mathbf{\xi}_h \mathbf{c}^\top$  for the unscaled case) are computed at each step  $h$ , where  $b_h = \omega_h^\top \mathbf{\xi}_h / (\mathbf{\xi}_h^\top \mathbf{\xi}_h)$  is the coefficient when you regress  $\omega_h$  on  $\mathbf{\xi}_h$  (and is at the core of the inner relation explicited in the next subsection). One can show that  $b_h = p_h$  and that  $\hat{\mathbf{Y}}_{(h)} = \mathcal{P}_{\mathbf{\xi}_h} \mathbf{Y}_{h-1}$ . Indeed, since  $\mathbf{v}_h = \mathbf{Y}_{h-1}^\top \mathbf{\xi}_h / \|\mathbf{Y}_{h-1}^\top \mathbf{\xi}_h\|$ , we obtain

$$\begin{aligned} \omega_h^\top \mathbf{\xi}_h &= p_h \alpha_h \mathbf{v}_h^\top \mathbf{Y}_{h-1}^\top \mathbf{\xi}_h \\ &= p_h \alpha_h \|\mathbf{Y}_{h-1}^\top \mathbf{\xi}_h\| \\ &= p_h \|\mathbf{\xi}_h\|^2. \end{aligned}$$

For scaled weights, we have

$$\begin{aligned} \hat{\mathbf{Y}}_{(h)} &= b_h \mathbf{\xi}_h \mathbf{v}_h^\top \\ &= \frac{\|\mathbf{Y}_{h-1}^\top \mathbf{\xi}_h\|}{(\mathbf{\xi}_h^\top \mathbf{\xi}_h)} \mathbf{\xi}_h \frac{\mathbf{\xi}_h^\top \mathbf{Y}_{h-1}}{\|\mathbf{Y}_{h-1}^\top \mathbf{\xi}_h\|} \\ &= \mathbf{\xi}_h (\mathbf{\xi}_h^\top \mathbf{\xi}_h)^{-1} \mathbf{\xi}_h^\top \mathbf{Y}_{h-1} \\ &= \mathcal{P}_{\mathbf{\xi}_h} \mathbf{Y}_{h-1}. \end{aligned}$$

For unscaled weights, we have also

$$\hat{\mathbf{Y}}_{(h)} = b_h \mathbf{\xi}_h \mathbf{c}^\top = 1 \cdot \mathbf{\xi}_h (\mathbf{\xi}_h^\top \mathbf{\xi}_h)^{-1} \mathbf{\xi}_h^\top \mathbf{Y}_{h-1} = \mathcal{P}_{\mathbf{\xi}_h} \mathbf{Y}_{h-1}.$$

As in case (ii), using [31, Theorem 7, p. 151], we obtain

$$\hat{\mathbf{Y}}_{(h)} = \mathcal{P}_{\mathbf{\xi}_h} \mathbf{Y}_{h-1} = \mathcal{P}_{\mathbf{\xi}_h} (\mathbf{I} - \mathcal{P}_{\mathbf{\Xi}_{\bullet, h-1}}) \mathbf{Y} = \mathcal{P}_{\mathbf{\xi}_h} \mathbf{Y}$$

and

$$\hat{\mathbf{Y}}_h = \sum_{j=1}^h \hat{\mathbf{Y}}_{(j)} = \mathcal{P}_{\mathbf{\Xi}_{\bullet, h}} \mathbf{Y}.$$

### E. Proof of the inner relation in PLS-R

The central inner PLS relation is made of successive *univariate* regressions of  $\omega_h$  upon  $\mathbf{\xi}_h$ . This constitute the link between  $Y$  and  $X$  in the PLS model. This link is estimated one dimension at a time (partial modeling) hence the original ‘‘Partial’’ in the PLS acronym.

We have

$$\begin{aligned} \omega_h &= \mathcal{P}_{\mathbf{\xi}_h} \omega_h + \mathcal{P}_{\mathbf{\xi}_h^\perp} \omega_h \\ &= \mathbf{\xi}_h (\mathbf{\xi}_h^\top \mathbf{\xi}_h)^{-1} \mathbf{\xi}_h^\top \omega_h + \mathcal{P}_{\mathbf{\xi}_h^\perp} \omega_h \\ &= \mathbf{\xi}_h p_h + \mathcal{P}_{\mathbf{\xi}_h^\perp} \omega_h \\ &:= \mathbf{\xi}_h p_h + \mathbf{r}_h. \end{aligned}$$

This leads to

$$\mathbf{\Omega}_{\bullet, h} = \mathbf{\Xi}_{\bullet, h} \mathbf{P}_h + \mathbf{R}_{\bullet, h},$$

where  $\mathbf{P}_h = \text{diag}(p_j)_{1 \leq j \leq h}$  and  $\mathbf{R}_H = [\mathbf{r}_1, \dots, \mathbf{r}_H]$ .



### F. The decomposition model in PLS-R

Note that due to the properties on the  $\xi_j$ , we have that  $\Xi_H^\top \Xi_H$  is a diagonal matrix and also that  $\xi_h^\top = \xi_h^\top \left( \prod_{j=h-1}^1 \mathcal{P}_{\xi_j^\perp} \right) = \xi_h^\top \mathcal{P}_{\Xi_{\bullet, h-1}^\perp}$ . This allows us to write

$$\begin{aligned} C_H^\top &:= (\Xi_H^\top \Xi_H)^{-1} \Xi_H^\top X \\ &= (\Xi_H^\top \Xi_H)^{-1} \begin{bmatrix} \xi_1^\top \\ \xi_2^\top \mathcal{P}_{\Xi_{\bullet, 1}^\perp} \\ \vdots \\ \xi_H^\top \mathcal{P}_{\Xi_{\bullet, H-1}^\perp} \end{bmatrix} X \\ &= (\Xi_H^\top \Xi_H)^{-1} \begin{bmatrix} \xi_1^\top X_0 \\ \xi_2^\top X_1 \\ \vdots \\ \xi_H^\top X_{H-1} \end{bmatrix} \\ &= \begin{bmatrix} (\xi_1^\top \xi_1)^{-1} \xi_1^\top X_0 \\ (\xi_2^\top \xi_2)^{-1} \xi_2^\top X_1 \\ \vdots \\ (\xi_H^\top \xi_H)^{-1} \xi_H^\top X_{H-1} \end{bmatrix}, \end{aligned}$$

Since  $v_h$  is normed, then  $\mathcal{P}_{v_h} = v_h v_h^\top$ . Now, looking more closely at the PLS-R algorithm (see, e.g., [63, p. 3], [64]), it is clear that

$$\omega_h = Y_{h-1} v_h$$

and

$$Y_{h-1} = \mathcal{P}_{\xi_h^\perp} Y_{h-1} + \mathcal{P}_{\xi_h} Y_{h-1} = Y_h + \mathcal{P}_{\xi_h} Y_{h-1}.$$

We thus have

$$\begin{aligned} Y_{h-1} &= Y_{h-1} \mathcal{P}_{v_h} + Y_{h-1} \mathcal{P}_{v_h^\perp} \\ &= \omega_h v_h^\top + Y_{h-1} - Y_{h-1} \mathcal{P}_{v_h} \\ &= \omega_h v_h^\top + Y_h + \mathcal{P}_{\xi_h} Y_{h-1} - Y_{h-1} \mathcal{P}_{v_h}. \end{aligned}$$

By recurrence, we obtain

$$\begin{aligned} Y_0 &= \sum_{j=1}^H [\omega_j v_j^\top + \mathcal{P}_{\xi_j} Y_{j-1} - Y_{j-1} \mathcal{P}_{v_j}] + Y_H \\ &= \Omega_H D_H^\top + \sum_{j=1}^H [\xi_j (\xi_j^\top \xi_j)^{-1} \xi_j^\top Y_{j-1} - Y_{j-1} v_j v_j^\top] + Y_H \quad \text{so} \\ &= \Omega_H D_H^\top + \sum_{j=1}^H [\xi_j g_j^\top - \omega_j v_j^\top] + Y_H \\ &= \Omega_H D_H^\top + \Xi_H G_H^\top - \Omega_H D_H^\top + Y_H, \end{aligned}$$

where we have defined  $D_H = [v_1, \dots, v_H]$  and where

$$G_H^\top := \begin{bmatrix} (\xi_1^\top \xi_1)^{-1} \xi_1^\top Y_0 \\ (\xi_2^\top \xi_2)^{-1} \xi_2^\top Y_1 \\ \vdots \\ (\xi_H^\top \xi_H)^{-1} \xi_H^\top Y_{H-1} \end{bmatrix}.$$

**Remark 7.** Let

$$\begin{aligned} \tilde{G}_H^\top &:= \begin{bmatrix} (\|Y_0^\top \xi_1\|)^{-1} \xi_1^\top Y_0 \\ (\|Y_1^\top \xi_2\|)^{-1} \xi_2^\top Y_1 \\ \vdots \\ (\|Y_{H-1}^\top \xi_H\|)^{-1} \xi_H^\top Y_{H-1} \end{bmatrix} \\ &= P_H^{-1} G_H^\top, \end{aligned}$$

where  $P_H = (p_h)$  is the diagonal matrix defined in subsection A-E. It can be seen to have entries  $\|Y_{h-1}^\top \xi_h\| / (\xi_h^\top \xi_h)$ ,  $h = 1, \dots, H$ . We have

$$\begin{aligned} Y_h &= Y_{h-1} - \xi_h g_h^\top \\ &= Y_{h-1} - p_h \xi_h \tilde{g}_h^\top, \end{aligned}$$

the first deflation step formula being the one used in [42] while the second is the one used in [41]; see also [63].

We have [42, p. 101, g)]

$$\begin{aligned} X_h &= \mathcal{P}_{\xi_h^\perp} X_{h-1} \\ &= (I - \xi_h (\xi_h^\top \xi_h)^{-1} \xi_h^\top) X_{h-1} \\ &= (I - X_{h-1} u_h (\xi_h^\top \xi_h)^{-1} \xi_h^\top) X_{h-1} \\ &= X_{h-1} (I - u_h (\xi_h^\top \xi_h)^{-1} \xi_h^\top X_{h-1}) \\ &= X_{h-2} (I - u_{h-1} (\xi_{h-1}^\top \xi_{h-1})^{-1} \xi_{h-1}^\top X_{h-2}) \\ &\quad \times (I - u_h (\xi_h^\top \xi_h)^{-1} \xi_h^\top X_{h-1}) \\ &= X_0 \prod_{j=1}^h (I - u_j (\xi_j^\top \xi_j)^{-1} \xi_j^\top X_{j-1}) \\ &= X_0 \prod_{j=1}^h (I - u_j (\xi_j^\top \xi_j)^{-1} \xi_j^\top \mathcal{P}_{\Xi_{\bullet, j-1}^\perp} X_0) \\ &= X_0 \prod_{j=1}^h (I - u_j (\xi_j^\top \xi_j)^{-1} \xi_j^\top (I - \mathcal{P}_{\Xi_{\bullet, j-1}}) X_0) \\ &= X_0 \prod_{j=1}^h (I - u_j (\xi_j^\top \xi_j)^{-1} \xi_j^\top X_0) \\ &:= X_0 A^{(h)} \end{aligned}$$

$$\begin{aligned} \Xi_{\bullet, h} &= [\xi_1, \dots, \xi_h] \\ &= [X u_1, \dots, X_{h-1} u_h] \\ &= [X u_1, \dots, X A^{(h-1)} u_h] \\ &= X [u_1, \dots, A^{(h-1)} u_h] \end{aligned}$$

Let the matrix of adjusted weights be  $\tilde{W}_{\bullet, h} = [\tilde{w}_1, \dots, \tilde{w}_h]$  with  $\tilde{w}_1 = u_1$  and  $\tilde{w}_h = \prod_{j=1}^{h-1} (I - u_j c_j^\top) u_h = \prod_{j=1}^{h-1} (I - u_j (\xi_j^\top \xi_j)^{-1} \xi_j^\top X) u_h = A^{(h-1)} u_h$ . It is thus clear that  $X w_h = X_{h-1} u_h$  and  $\Xi_{\bullet, h} = X \tilde{W}_{\bullet, h}$ ; see [42, p. 135]. Interestingly, from [42, p. 114], we can also write  $\tilde{W}_{\bullet, h} = U_{\bullet, h} (C_{\bullet, h}^\top U_{\bullet, h})^{-1}$ .

From [31, Theorem 4, p. 106],

$$\begin{aligned} C_H^\top \tilde{W}_H C_H^\top &= (\Xi_H^\top \Xi_H)^{-1} \Xi_H^\top X \tilde{W}_H C_H^\top \\ &= (\Xi_H^\top \Xi_H)^{-1} \Xi_H^\top \Xi_H C_H^\top \\ &= C_H^\top \end{aligned}$$

so that  $\tilde{W}_H$  is a generalised inverse of  $C_H^\top$  [37].

Suppose that  $\text{rank}(X_0) = r \leq p$ . We have  $\xi_1 = X_0 u_1 \in \mathcal{I}(X)$  of dimension  $r$  (as a combination of the columns of  $X_0$ ). Then we define  $X_1 = \mathcal{P}_{\xi_1^\perp} X_0 = \mathcal{P}_{\xi_1^\perp \cap \mathcal{I}(X)} X_0 + \mathcal{P}_{\xi_1^\perp \cap \mathcal{I}(X)^\perp} X_0 = \mathcal{P}_{\xi_1^\perp \cap \mathcal{I}(X)} X_0$ . So the columns of  $X_1$  belong to  $\mathcal{I}(X) \cap \{\xi_1\}^\perp$ , which is of dimension  $r - 1$ . We iterate this process [3, Sec. 5] until we obtain  $X_r$  which will be of rank 0 (and so  $X_r = 0$ ). We thus have the (exact) decomposition when  $H = r$ :

$$X = \Xi_{\bullet r} C_{\bullet r}^\top.$$

From [65, eq. 2.22 p. 16], the columns of  $\Xi_{\bullet r}$  are linearly independent. From [65, eq. 7.54(d) p. 139],  $\Xi_{\bullet r}^+ = (\Xi_{\bullet r}^\top \Xi_{\bullet r})^{-1} \Xi_{\bullet r}^\top$  and  $\Xi_{\bullet r} \Xi_{\bullet r}^+ = I_r$ . So, we obtain

$$\Xi_{\bullet r}^+ X = C_{\bullet r}^\top.$$

### G. The adjusted weight optimisation problem

Until now we have defined the X-scores  $\xi_h$  in terms of the deflated matrix  $X_h$ , however, we can also define the scores using the original matrix  $X$  by a set of adjusted weight vectors  $\tilde{w}_h$  [40], as proved in the previous subsection:

$$\xi_h = X \tilde{w}_h = X_{h-1} u_h, \quad h = 1, \dots, H. \quad (16)$$

Let  $\tilde{W}_{\bullet h}$  denote the matrix with column vectors  $\tilde{w}_1, \dots, \tilde{w}_h$  so that,

$$\begin{aligned} X &= \Xi_{\bullet h} C_{\bullet h}^\top + X_h \\ &= X \tilde{W}_{\bullet h} C_{\bullet h}^\top + X_h. \end{aligned}$$

Using the definition (16) for any  $h > 1$ , and rearranging the above decomposition, we can write:

$$\xi_h = X_{h-1} u_h = X(I_p - \tilde{W}_{\bullet h-1} C_{\bullet h-1}^\top) u_h$$

and thus we can define the adjusted weights as:

$$\tilde{w}_h = (I_p - \tilde{W}_{\bullet h-1} C_{\bullet h-1}^\top) u_h. \quad (17)$$

Thus the adjusted weights can be found using the loadings and weights from previous iterations. Rearranging for  $u_h$  we have,

$$\begin{aligned} u_h &= \tilde{w}_h + \tilde{W}_{\bullet h-1} C_{\bullet h-1}^\top u_h \\ &= \tilde{w}_h - \tilde{W}_{\bullet h-1} g_h \end{aligned} \quad (18)$$

where  $g_h = -C_{\bullet h-1}^\top u_h$ .

We have seen that  $\tilde{W}_{\bullet h} = U_{\bullet h} (C_{\bullet h}^\top U_{\bullet h})^{-1}$ , so that  $\tilde{W}_{\bullet h}^\top \tilde{W}_{\bullet h} = (C_{\bullet h}^\top U_{\bullet h} U_{\bullet h}^\top C_{\bullet h})^{-1}$  and  $(\tilde{W}_{\bullet h}^\top \tilde{W}_{\bullet h})^{-1}$  exists. Consequently,  $\tilde{W}_{\bullet h}^+ = (\tilde{W}_{\bullet h}^\top \tilde{W}_{\bullet h})^{-1} \tilde{W}_{\bullet h}^\top = C_{\bullet h}^\top U_{\bullet h} U_{\bullet h}^\top$  and  $\tilde{W}_{\bullet h}^+ \tilde{W}_{\bullet h} = I$ .

To express  $\tilde{w}_h$  in terms of  $u_h$  we first note that,

$$\tilde{W}_{\bullet h-1} g_h = \tilde{w}_h - u_h,$$

so that

$$\begin{aligned} g_h &= \tilde{W}_{\bullet h-1}^+ (w_h - u_h) \\ &= \tilde{W}_{\bullet h-1}^+ w_h \end{aligned} \quad (19)$$

where we use the fact that  $\tilde{W}_{\bullet h-1}^+ u_h = 0_{h-1}$  (since  $U_{\bullet h-1}^\top u_h = 0$ ).

Combining equations (19) and (18) gives:

$$\begin{aligned} u_h &= (I_p - \tilde{W}_{\bullet h-1} \tilde{W}_{\bullet h-1}^+) \tilde{w}_h \\ &= \mathcal{P}_{\tilde{W}_{\bullet h-1}^\perp} \tilde{w}_h. \end{aligned}$$

### H. The sparse PLS weights

The optimisation function for the  $\tilde{u}$  in sparse PLS is:

$$\tilde{u}_h = \underset{\tilde{u}}{\text{argmin}} \left\{ \|\mathbf{M}_{h-1} - \tilde{u} \mathbf{v}^\top\|_F^2 + 2\lambda_1 \|\tilde{u}\|_1 \right\}. \quad (20)$$

We denote  $m_{ij,h}$  the entry  $(i, j)$  of  $\mathbf{M}_h$ ,  $h = 1, \dots, H$ . Solving this problem, we rewrite the criterion (20) as a separable function

$$\sum_{i=1}^p \left\{ \sum_{j=1}^q (m_{ij} - \tilde{u}_i v_j)^2 + 2\lambda_1 |\tilde{u}_i| \right\}.$$

Therefore, we can optimise over individual components of  $\tilde{u}$  separately. Expanding the squares and observing that  $\|\mathbf{v}\|_2 = 1$ , we obtain

$$\begin{aligned} \sum_{j=1}^q (m_{ij} - \tilde{u}_i v_j)^2 &= \sum_{j=1}^q m_{ij}^2 - 2 \sum_{j=1}^q m_{ij} \tilde{u}_i v_j + \sum_{j=1}^q \tilde{u}_i^2 v_j^2 \\ &= \sum_{j=1}^q m_{ij}^2 - 2(\mathbf{M} \mathbf{v})_i \tilde{u}_i + \tilde{u}_i^2, \end{aligned}$$

where  $\mathbf{M}_h = (m_{ij})$ . Hence, the optimal  $\tilde{u}_i$  minimises  $\tilde{u}_i^2 - 2(\mathbf{M} \mathbf{v})_i \tilde{u}_i + 2\lambda_1 |\tilde{u}_i|$ . By using [30, Lemma 2], we find

$$\tilde{u}_i = g^{\text{soft}}((\mathbf{M} \mathbf{v})_i, \lambda_1).$$

Similarly, optimisation over  $\tilde{v}$  for a fixed (normed)  $\mathbf{u}$  is also obtained by optimising over individual components:

$$\tilde{v}_j = g^{\text{soft}}((\mathbf{M}^\top \mathbf{u})_j, \lambda_2).$$

The minimiser of (20) is obtained by applying the thresholding function  $g^{\text{soft}}(\cdot, \lambda)$  to the vector  $\mathbf{M} \mathbf{v}$  componentwise and to the vector  $\mathbf{M}^\top \mathbf{u}$  componentwise too.

### REFERENCES

- [1] H. Wold, "Estimation of principal components and related models by iterative least squares," in *Multivariate Analysis*. Dayton, Ohio: Academic Press, New York, Wiley, June 1966, pp. 391–420.
- [2] A. Krishnan, L. J. Williams, A. R. McIntosh, and H. Abdi, "Partial least squares (pls) methods for neuroimaging: A tutorial and review," *NeuroImage*, vol. 56, no. 2, pp. 455 – 475, 2011.
- [3] H. Abdi and L. Williams, *Partial Least Squares Methods: Partial Least Squares Correlation and Partial Least Squares Regression*, ser. Methods in Molecular Biology. Springer, 2012, vol. 930, ch. 23, pp. 549–579.
- [4] F. J. Rohlf and M. Corti, "Use of two-block partial least-squares to study covariation in shape," *Systematic Biology*, vol. 49, no. 4, pp. 740–753, 2000.
- [5] V. Vinzi, L. Trinchera, and S. Amato, "Pls path modeling: from foundations to recent developments and open issues for model assessment and improvement," *Handbook of Partial Least Squares*, pp. 47–82, 2010.

- [6] A. D. Cak, E. F. Moran, R. de O. Figueiredo, D. Lu, G. Li, and S. Hetrick, "Urbanization and small household agricultural land use choices in the Brazilian Amazon and the role for the water chemistry of small streams," *Journal of Land Use Science*, vol. 11, no. 2, pp. 203–221, 2016.
- [7] J. A. Wegelin, "A survey of partial least squares (pls) methods, with emphasis on the two-block case," University of Washington, Tech. Rep., 2000.
- [8] G. Guo and G. Mu, "Joint estimation of age, gender and ethnicity: Cca vs. pls," in *10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, April 2013, pp. 1–6.
- [9] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: an overview with application to learning methods," *Neural Computation*, vol. 16, no. 12, pp. 2639–2664, 2004.
- [10] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, no. 3-4, p. 321, 1936.
- [11] S. Wold, M. Sjström, and L. Eriksson, "Pls-regression: a basic tool of chemometrics," *Chemometrics and Intelligent Laboratory Systems*, vol. 58, no. 2, pp. 109 – 130, 2001.
- [12] R. Rosipal and N. Krämer, "Overview and recent advances in partial least squares," in *Subspace, Latent Structure and Feature Selection: Statistical and Optimization Perspectives Workshop*, February 2006, pp. 34–51.
- [13] P. Geladi and B. R. Kowalski, "Partial least-squares regression: a tutorial," *Analytica Chimica Acta*, vol. 185, pp. 1–17, 1986.
- [14] A.-L. Boulesteix and K. Strimmer, "Partial least squares: a versatile tool for the analysis of high-dimensional genomic data," *Briefings in Bioinformatics*, vol. 8, no. 1, pp. 32–44, 2007.
- [15] G. Ji, Z. Yang, and W. You, "Pls-based gene selection and identification of tumor-specific genes," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 6, pp. 830–841, 2011.
- [16] B. Lique, P. Lafaye de Micheaux, B. Hejblum, and R. Thiébaud, "Group and sparse group partial least square approaches applied in genomics context," *Bioinformatics*, vol. 32, pp. 35–42, 2016.
- [17] A. R. McIntosh, F. L. Bookstein, J. V. Haxby, and C. L. Grady, "Spatial pattern analysis of functional brain images using partial least squares," *NeuroImage*, vol. 3, no. 3, pp. 143–157, 1996.
- [18] P. V. Roon, J. Zakizadeh, and S. Chartier, "Partial least squares tutorial for analyzing neuroimaging data," *The Quantitative Methods for Psychology*, vol. 10, no. 2, pp. 200–215, 2014.
- [19] M. Lorenzi, B. Gutman, D. P. Hibar, A. Altmann, N. Jahanshad, P. M. Thompson, and S. Ourselin, "Partial least squares modelling for imaging-genetics in Alzheimer's disease: Plausibility and generalization," in *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, April 2016, pp. 838–841.
- [20] J. Liu and V. D. Calhoun, "A review of multivariate analyses in imaging genetics," *Frontiers in Neuroinformatics*, vol. 8, no. 29, 2014.
- [21] K.-A. Lê Cao, D. Rossouw, C. Robert-Granié, and P. Besse, "Sparse PLS: Variable Selection when Integrating Omics data," *Statistical Application and Molecular Biology*, vol. 7, no. (1):37, 2008.
- [22] C. Dhanjal, S. R. Gunn, and J. Shawe-Taylor, "Efficient sparse kernel feature extraction based on partial least squares," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 8, pp. 1347–1361, Aug 2009.
- [23] D. M. Witten, R. Tibshirani, and T. Hastie, "A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis," *Biostatistics*, vol. 10, no. 3, pp. 515–534, 2009.
- [24] D. Chung and S. Keleş, "Sparse Partial Least Squares Classification for High Dimensional Data," *Statistical Applications in Genetics and Molecular Biology*, vol. 9, no. 1, p. 17, 2010.
- [25] H. Chun and S. Keleş, "Sparse partial least squares regression for simultaneous dimension reduction and variable selection," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 1, pp. 3–25, 2010.
- [26] D. Lin, H. Cao, V. D. Calhoun, and Y.-P. Wang, "Sparse models for correlative and integrative analysis of imaging and genetic data," *Journal of Neuroscience Methods*, vol. 237, pp. 69 – 78, 2014.
- [27] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2015. [Online]. Available: <https://www.R-project.org/>
- [28] H. Lütkepohl, *New introduction to multiple time series analysis*. Berlin: Springer-Verlag, 2005.
- [29] D. A. Harville, *Matrix Algebra From a Statistician's Perspective*. Springer, 1997.
- [30] H. Shen and J. Z. Huang, "Sparse principal component analysis via regularized low rank matrix approximation," *Journal of Multivariate Analysis*, vol. 99, no. 6, pp. 1015 – 1034, 2008.
- [31] S. Puntanen, G. Styan, and J. Isotalo, *Matrix Tricks for Linear Statistical Models - Our Personal Top Twenty*. Springer-Verlag Berlin Heidelberg 2011, 2011.
- [32] K. V. Mardia, J. T. Kent, and J. M. Bibby, *Multivariate analysis / K.V. Mardia, J.T. Kent, J.M. Bibby*. Academic Press London ; New York, 1979.
- [33] F. A. Nielsen, "Neuroinformatics in functional neuroimaging," Ph.D. dissertation, Technical University of Denmark, Lyngby, 2002.
- [34] H. Vinod, "Canonical ridge and econometrics of joint production," *Journal of Econometrics*, vol. 4, no. 2, pp. 147 – 166, 1976.
- [35] B. W. S. S. E. Leurgans, R. A. Moyeed, "Canonical correlation analysis when the data are curves," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 55, no. 3, pp. 725–740, 1993.
- [36] S. Wold, A. Ruhe, H. Wold, and W. J. Dunn, "The collinearity problem in linear regression. the partial least squares (pls) approach to generalized inverses," *SIAM Journal on Scientific and Statistical Computing*, vol. 5, no. 3, pp. 735–743, 1984.
- [37] S. de Jong, "Simpls: an alternative approach to partial least squares regression," *Chemometrics and Intelligent Laboratory Systems*, vol. 18, pp. 251–263, 1993.
- [38] F. Lindgren and S. Rännar, "Alternative partial least squares (pls) algorithms," *Perspectives Drug Discovery and Design*, pp. 105–113, 1998.
- [39] A. Alin, "Comparison of pls algorithms when number of objects is much larger than number of variables," *Statistical Papers*, vol. 50, pp. 711–720, 2009.
- [40] C. J. F. ter Braak and S. de Jong, "The objective function of partial least squares regression," *Journal of Chemometrics*, vol. 12, no. 1, pp. 41–54, 1998.
- [41] A. Höskuldsson, "Pls regression methods," *Journal of Chemometrics*, vol. 2, pp. 211–228, 1988.
- [42] M. Tenenhaus, *La régression PLS: Théorie et Pratique*. Paris: Technip, 1998.
- [43] A. Phatak and S. De Jong, "The geometry of partial least squares," *Journal of Chemometrics*, vol. 11, no. 4, pp. 311–338, 1997.
- [44] I. Frank and J. Friedman, "A statistical view of some chemometrics regression tools," *Technometrics*, pp. 109–135, 1993.
- [45] A. Burnham and R. Viveros, "Frameworks for latent variable multivariate regression," *Journal of Chemometrics*, vol. 10, pp. 31–45, 1996.
- [46] E. Zhu and R. Barnes, "A simple iteration algorithm for pls regression," *Journal of Chemometrics*, vol. 9, pp. 363–372, 1995.
- [47] L. Mackey, "Deflation methods for sparse pca," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., 2009, pp. 1017–1024.
- [48] G. I. Allen, C. Peterson, M. Vannucci, and M. Maletic-Savatic, "Regularized Partial Least Squares with an Application to NMR Spectroscopy," *Statistical Analysis and Data Mining*, vol. 6, no. 4, pp. 302–314, Aug 2013.
- [49] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B*, vol. 58, pp. 267–288, 1994.
- [50] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.
- [51] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, "A sparse-group lasso," *Journal of Computational and Graphical Statistics*, vol. 22, no. 2, pp. 231–245, 2013.
- [52] X. Chen and H. Liu, "An efficient optimization algorithm for structured sparse cca, with applications to eqtl mapping," *Statistics in Biosciences*, vol. 4, no. 1, pp. 3–26, 2012.
- [53] M. Sutton, T. R., and B. Lique, "parse group subgroup partial least squares with application to genomics data," *to appear*, 2017.
- [54] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, "Sparsity and smoothness via the fused lasso," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 1, pp. 91–108, 2005.
- [55] J. Baglama and L. Reichel, *irlba: Fast Truncated SVD, PCA and Symmetric Eigendecomposition for Large Dense and Sparse Matrices*, 2015, r package version 2.0.0. [Online]. Available: <http://CRAN.R-project.org/package=irlba>
- [56] —, "Augmented implicitly restarted lanczos bidiagonalization methods," *SIAM Journal on Scientific Computing*, vol. 27, no. 1, pp. 19–42, 2005.

- [57] F. Liang, R. Shi, and Q. Mo, "A split-and-merge approach for singular value decomposition of large-scale matrices," *Statistics And Its Interface*, vol. 9, no. 4, pp. 453–459, 2016.
- [58] H. Cardot and D. Degras, "Online Principal Component Analysis in High Dimension: Which Algorithm to Choose?" *ArXiv e-prints*, submitted for publication.
- [59] D. Nguyen and D. Rocke, "Tumor classification by partial least squares using microarray gene expression data," *Bioinformatics*, vol. 18, no. 1, pp. 39–50, 2002.
- [60] M. Barker and W. Rayens, "Partial least squares for discrimination," *Journal of Chemometrics*, vol. 17, no. 3, pp. 166–173, 2003.
- [61] R. J. Tibshirani and J. Taylor, "The solution path of the generalized lasso," *Annals of Statistics*, vol. 39, no. 3, pp. 1335–1371, 2011.
- [62] W. W. Hsieh, *Machine Learning Methods in the Environmental Sciences*. New York, NY, USA: Cambridge University Press, 2009.
- [63] V. Esposito Vinzi and G. Russolillo, "Partial least squares algorithms and methods," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 5, no. 1, pp. 1–19, 2013.
- [64] H. Abdi, *Encyclopedia of Measurement and Statistics*. Thousand Oaks (CA): Sage, 2007, ch. Partial Least Square Regression.
- [65] G. A. F. Seber, *A matrix handbook for statisticians*, ser. Wiley Series in Probability and Statistics. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, 2008.